VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2019 Bc. Martin Bubeník

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

RASPBERRYPI KAMEROVÝ CHECKER
RASPBERRYPI CAMERA CHECKER

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Martin Bubeník**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Peter Honec, Ph.D.**
SUPERVISOR

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Kybernetika, automatizace a měření**
Ústav automatizace a měřicí techniky

Student: Bc. Martin Bubeník ***ID:*** 164842
Ročník: 2 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

RaspberryPI kamerový checker

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je navržení vybrané kamerové průmyslové inspekce konektoru na platformě RaspberryPI. Aplikace by měla splňovat standardní průmyslové požadavky (zobrazení a uložení výsledku, komunikace s PLC, výběr receptrury,...) Součástí práce je výběr vhodné kamery, optiky a osvětlení. Předpokládaný použitý programovací jazyk je Python s knihovnami OpenCV.

1. Seznamte se řetězcem strojového vidění a aplikací strojového vidění v průmyslu.
2. Seznámte se s vývojem aplikací v jazyce Python na platformě Raspberry PI a knihovnou OpenCV.
3. Vyberte vhodný kamerový modulu, optiku a osvětlení s ohledem na typ úlohy/měřené součástky.
4. Navrhněte mechaniku inspekce s využitím 3D tisku (lůžko, držák, podpůrné mechanické prvky).
5. Navrhněte aplikaci s definovanými požadavky (zobrazení a ukládání výsledků, GUI, DIO,...)
6. Zhodnoťte řešení (omezení, rychlost, vylepšení...)

DOPORUČENÁ LITERATURA:

ASHWIN PAJANKAR: Raspberry Pi Computer Vision Programming, ISBN 978-1784398286

HALFACREE GARET, UPTON EBEN: Raspberry Pi: User Guide 3rd Edition, ISBN 978-80-251-4819-8

HLAVAC V., SONKA M., BOYLE R.: Image Processing, Analysis, and Machine Vision, ISBN 978-0495082521

Termín zadání: 4.2.2019 ***Termín odevzdání:*** 13.5.2019

Vedoucí práce: Ing. Peter Honec, Ph.D.
Konzultant:

doc. Ing. Václav Jirsík, CSc.
předseda oborové rady

ABSTRAKT

Diplomová práca sa venuje priemyselnej inšpekcií výrobných čísel konektorov na základe počítačového rozpoznávania, pričom aplikácia pre detekciu a rozpoznávanie je implementovaná v jazyku Python na platforme Raspberry Pi. Ku rozpoznávaniu práca využíva empiricky známu knižnicu OpenCV. Práca sa taktiež zaoberá výberom vhodných hardwarových prostriedkov, ktorými sú kamera s objektívom a osvetľovač, z ktorých je vytvorené spolu s mikropočítačom Raspberry Pi jedno kompaktné zariadenie. Kompaktné zariadenie je ďalej upevnené na navrhnutú mechanickú konštrukciu, pod ktorou je vytvorená inšpekčná zóna. Raspberry Pi nakoniec disponuje webovým užívateľským rozhraním pre kontrolu inšpekcie a rozhraním pre zápis získaných dát do databázy.

KĽÚČOVÉ SLOVÁ

Raspberry Pi, kamera, optika, osvetlenie, mechanika, Python, OpenCV, rozpoznávanie, GUI, databáza

ABSTRACT

The diploma thesis deals with the industrial inspection of correctly made connectors based on computer recognition, and the detection and recognition application is implemented in Python on the Raspberry Pi platform.The work uses empirically known OpenCV library for recognition. The work also deals with the selection of suitable hardware devices, which are a camera with a lens and an illuminator, from which is created one compact device together with the Raspberry Pi microcomputer. The compact device is further mounted on the designed mechanical structure under which is created inspection zone. Finally, Raspberry Pi has a web-based user interface to check the inspection and the interface to write the data to the database.

KEYWORDS

Raspberry Pi, camera, optics, lighting, mechanics, Python, OpenCV, recognition, GUI, database

BUBENÍK, Martin. *RaspberryPI kamerový checker*. Brno, 2019, 81 s. Diplomová práca. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedúci práce: Ing. Peter Honec, Ph.D.

VYHLÁSENIE

Vyhlasujem, že som svoju diplomovú prácu na tému „RaspberryPI kamerový checker“ vypracoval samostatne pod vedením vedúceho diplomovej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedenej diplomovej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto diplomovej práce som neporušil autorské práva tretích osôb, najmä som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákoníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora

POĎAKOVANIE

Rád by som sa poďakoval vedúcemu diplomovej práce pánovi Ing. Petrovi Honcovi, Ph.D. za odborné vedenie, konzultácie, trpezlivosť a podnetné návrhy k práci. Tiež by som sa chcel poďakovať firme TE Connectivity za poskytnuté príslušenstvo použité v práci.

Brno ....

podpis autora

Obsah

Zoznam obrázkov

Zoznam tabuliek

ÚVOD

Vizuálna detekcia a rozpoznávanie objektov patrí v poslednom desaťročí medzi najväčšie výzvy počítačového videnia. Potenciálne uplatnenie systémov detekcie a rozpoznávania objektov je veľmi široké, od bezpečnostných systémov (napr. identifikácia oprávnených osôb), cez medicínske techniky (napr. detekcia tumorov), priemyselné aplikácie (napr. vizuálna inšpekcia výrobkov), robotiku (napr. navigácia robota v nedostupnom teréne), až po rozšírenú realitu a mnohé ďalšie. Úloha automaticky vyhľadať objekt sa spravidla formuluje ako úloha segmentácie objektu, prípadne iba ako detekcia opísaného štvoruholníka alebo ako rozpoznanie samotného objektu. V priemyselnej praxi sa taktiež často stretávame s konkrétnou úlohou detekcie a rozpoznávania, a to konkrétne výrobných čísiel a šarží produktov. Jedná sa o vizuálnu kontrolu znakov, ktorá prestupuje celým výrobným procesom a separuje dobré kusy (často značené OK kusy) a zlé kusy (často značené NOK kusy). Takáto inšpekcia vo výrobe je žiaduca, lebo priaznivo pomáha zvýšiť celkovú efektivitu výrobného procesu a linky, na ktorej sa vyrába. Štatistiky ohľadne obrobkov sú potom ukladané do rôznych nadradených systémov, ako sú databázy alebo cloudy. Takýto celkový prehľad výroby a efektivity, automatického triedenia vyrobených kusov bez použitia ľudských zdrojov, iba pomocou automatického strojového videnia, napĺňa víziu Industry 4.0.

Diplomová práca sa venuje súpisu teoretických poznatkov, potrebných pre následnú prácu a s praktickým výberom hardwarových prostriedkov vlastnej aplikácie strojového videnia. Po teoretických kapitolách sa práca zaoberá opisom navrhnutých prostriedkov vlastnej aplikácie od mikropočítača Raspberry Pi, cez kamerový modul s objektívom, až po návrh osvetlenia a mechanickej konštrukcie. Vlastná aplikácia strojového videnia má za úlohu pomocou kamery a vhodného osvetlenia detegovať objekt, rozpoznať z jeho povrchu číslice, resp. čísla šarží. Objektami sú tri druhy produktov spoločnosti *TE Connectivity*. Po úspešnom rozpoznaní sa má vyhodnotiť správnosť číslic v Raspberry Pi a výsledok odoslať do aplikácie vizualizácie (GUI), ako nadradeného systému a uložiť do databázy. Práca sa zaoberá taktiež návrhom jednoduchej mechanickej konštrukcie prípravku, na ktorom je upevnený celý systém rozpoznávania.

1 STROJOVÉ VIDENIE

1.1 Definícia a história strojového videnia

Rozvoj výpočtovej techniky v 70. rokoch 20. storočia umožnil svojím technologickým pokrokom spracovanie a prenos dostatočného množstva dát pre prácu s obrazovou informáciou. Odozvou na tento pokrok bol vznik nového vedného odboru s názvom počítačové videnie. Tento odbor sa všeobecne zaoberá automatizovaním činnosti, ktorá vychádza z informácií získaných na základe rozboru obrazovej informácie nasnímanej snímacím prvkom - kamerou. Strojové videnie je potom využitie počítačového videnia v priemyselnej automatizácii, kde sa nezaoberá iba samotným získaním informácií z obrazu, ale aj prenosom do ostatných automatizovaných zariadení sprostredkujúcich výrobný proces. Prvé skutočné zariadenie pre priemyselnú aplikáciu bolo vyvinuté v *Massachusetts Institute of Technology*, kde systém strojového videnia ovládal robotické rameno. [1][2]

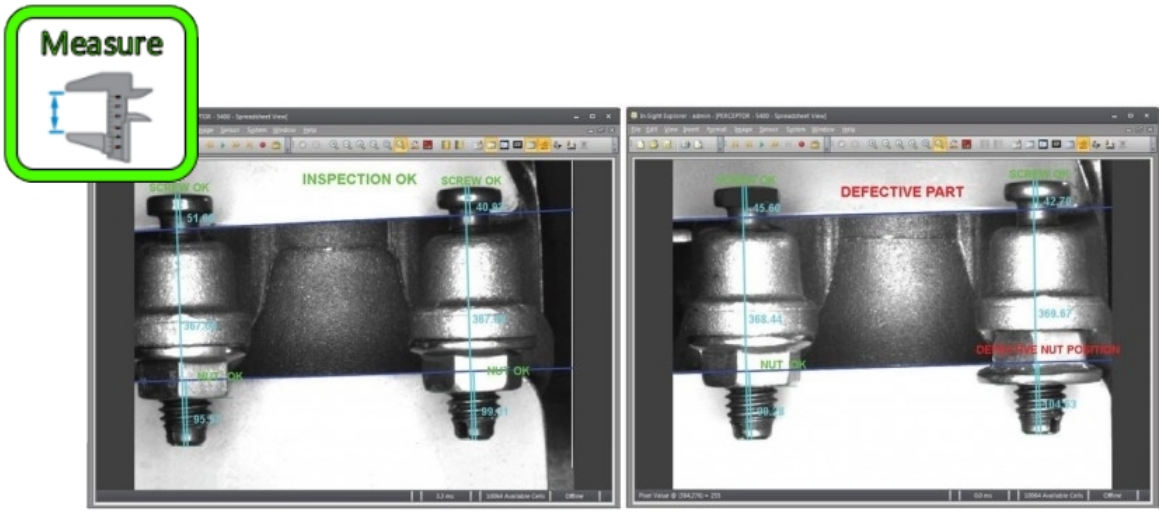
Ako každá priemyselná inovácia, tak aj strojové videnie sa začalo rozvíjať až keď došlo k synchronizácii dopytu s technickou realizáciou. Požiadavka po strojovom videní bola naštartovaná zmenou marketingových stratégií výrobných spoločností. Prelom 20. a 21. storočia je charakterizovaný preorientáciou firiem na marketingové riadenie, maximálnu kvalitu a uspokojenie požiadaviek zákazníka. Tento trend sa rozvíjal v priemysle v znamení dvoch hlavných požiadavok: zvyšovanie kvality a znižovanie nákladov. Dodávatelia sa teda začali obzerať po technických prostriedkoch, ktoré by im umožnili dokonale kontrolovať všetku produkciu. V rovnakej dobe boli uvedené na trh kvalitné CCD čipy pre snímanie obrazu, ktoré boli k dispozícii hlavne vďaka ich komerčnému využitiu v digitálnych fotoaparátoch. Zároveň sa tiež objavili rýchle procesory s vysokým výpočtovým výkonom za prijateľnú cenu. Strojové videnie tak získalo hnací motor k rozvoju a zároveň aj potrebné technické prostriedky. [3]

V dnešnej dobe ponúkajú výrobcovia ergonomické systémy so širokými možnosťami uplatnenia. Systémy sú komplexné a univerzálne, zvládajú bežne niekoľko štandardných aj neštandardných komunikačných protokolov, niektoré systémy sú dokonca obohatené aj o umelú inteligenciu. Zákazník môže zvoliť medzi komerčným riešením, ktoré ponúka hotový balíček pripravených služieb, alebo môže využiť službu s tzv. “open source” programom (“open source” znamená, že zdrojový kód je voľne prístupný každému, kto oň má záujem a môže ho upravovať podľa svojich predstáv, šíriť svoje distribúcie ďalej bez súhlasu autora), ktorý však často vyžaduje rozsiahle úpravy.

1.2 Aplikácia strojového videnia v priemysle

Pre operátorov výrobných liniek je pochopiteľne náročné upriamovať pozornosť a kontrolovať kvalitu súčiastok alebo iných aspektov, ktoré pozorujeme pri strojovom videní, počas celej pracovnej doby. Už po niekoľkých minútach sa človek nedokáže plne sústrediť na svoju pracovnú činnosť. Priemyselné kamerové systémy takéto úlohy zvládajú nepretržite s väčšou presnosťou, rýchlosťou, robustnosťou a opakovateľnosťou, čo patrí k hlavným výhodám aplikovania strojového videnia v priemysle. Vo výsledku to znamená, že vylučujú ľudský faktor a umožňujú zrýchlenie kontroly kvality, a teda možnosť zrýchlenia výrobného procesu či zvýšenia kvality vyrábaných produktov. [4]
Najčastejšie úlohy, ktoré plní strojové videnie v priemysle, sú:

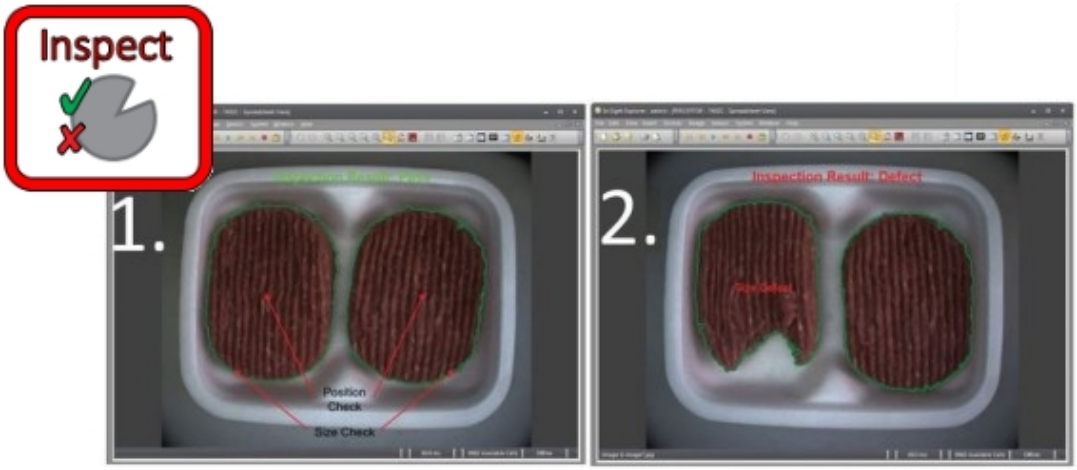
- **Meranie rozmerov a kontrola tolerancií** - úloha prestupuje celým procesom výroby, kľúčovou otázkou je možná dosiahnuteľnosť presnosti merania. Pre diely vyrobené tvárnením sú obvyklé tolerancie rozmerov v stotinách milimetra, pre diely vyrobené obrábaním sa tolerancie pohybujú v mikrometroch. Inšpekciu rozmerov dielu znázorňuje obrázok 1.1, pričom diel vľavo má všetky merané rozmery v tolerancii a diel vpravo nemá jeden meraný rozmer v tolerancii v mieste merania skrutky, pričom došlo k zámene s podložkou.



Obr. 1.1: Detekcia orientácie a rozmerov opisujúcich výrobok [4]

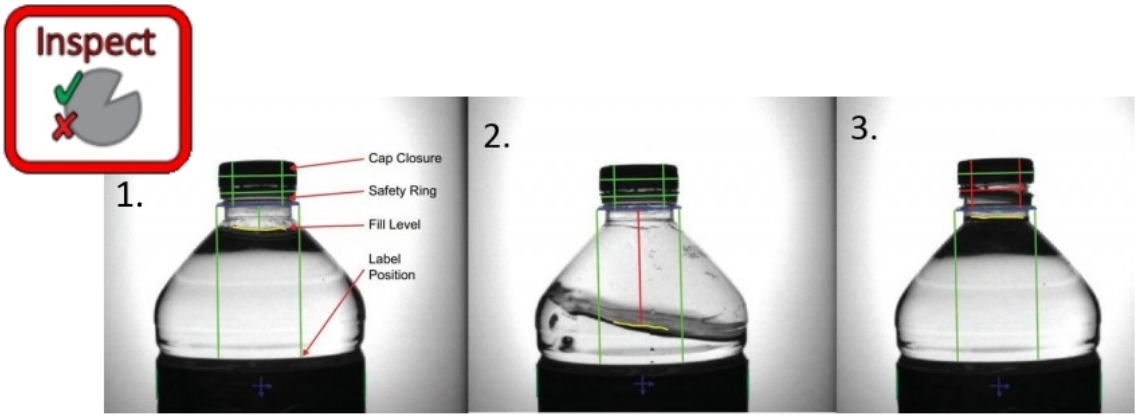
- **Detekcia vád opracovania** - úloha je typická pre fázu zhromaždenia výrobného materiálu k operácii, ktorá je sprevádzaná vstupnou kontrolou. Prítomná je takisto aj vo fáze výstupnej kontroly výsledného produktu. Môže sa jednať o detekciu poškodeniu povrchu, vady povrchových úprav alebo náterov, otrepov pri

strihaní materiálu, lisovanie, nedokonalé výlisky, deformácie, či rôzne poškodenia pri skladovaní alebo preprave. Vo výrobe zameranej na elektrotechniku je inšpekcia vykonávaná hlavne pri spájkovaní na plošnom spoji alebo pri usádzaní súčiastok. Inšpekciu vád opracovania znázorňuje obrázok 1.2, v ktorom výsledný produkt v potravinárskom priemysle vľavo je v poriadku kompletný, zatiaľ čo produkt vpravo je deformovaný a nevyhovujúci.



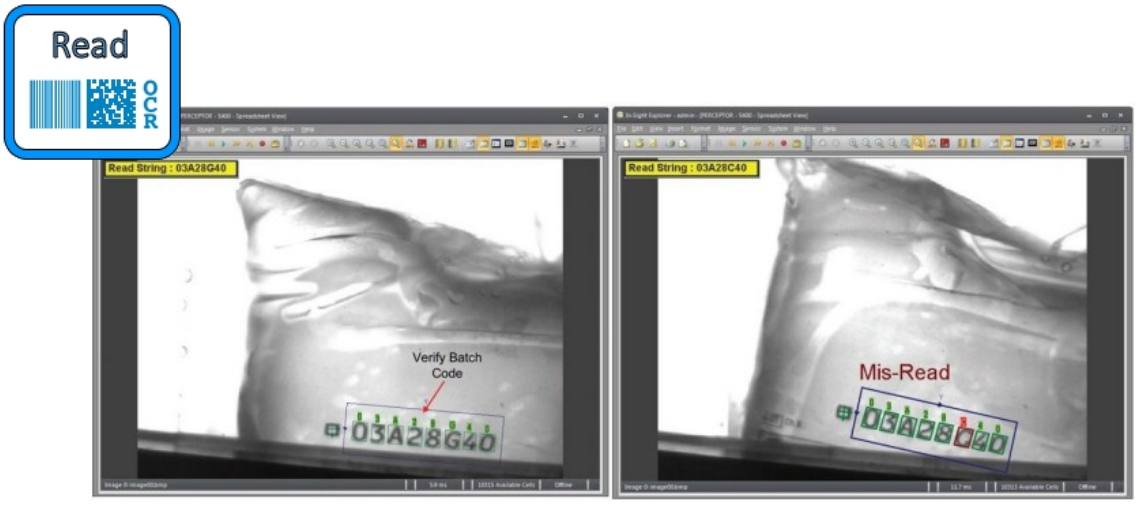
Obr. 1.2: Detekcia vád a deformácií objektu [4]

- **Počítanie, kontrola úplnosti a zabalenia** - úloha prestupuje celým procesom výroby, tento druh detekcie sa využíva v sériovej výrobe, kde ľudský faktor môže spôsobiť chybu a k zákazníkovi by sa dostal výrobok, ktorý nezodpovedá daným požiadavkám. Konkrétne sa využíva na kontrolu kvality a úplnosti produktov na výrobnej linke v potravinárskom priemysle k počítaniu komponentov, zisťovaniu chýbajúcich alebo poškodených produktov, ku kontrole hladiny vo fľaši alebo správnemu nasadeniu zátky alebo ku kontrole nepoškodenia produktov a obalových materiálov. Vo farmaceutickom priemysle sa používa na zistenie, či je v balení správny počet správnych tabliet. Inšpekciu hladiny vo fľaši a správnosť nasadenia uzáveru znázorňuje obrázok 1.3, na ktorom je vľavo správna výška hladiny vody a správne nasadený uzáver, v strede nesprávna výška hladiny vody a vpravo nesprávne nasadený uzáver.



Obr. 1.3: Detekcia úplnosti a zabalenia fľaše [4]

- **Čítanie textu** - čítanie textov je typické pre prvú fázu operácie, kedy je zisťované, či je k operácii pripravený správny diel na správnom mieste a pre identifikáciu jednotlivých dielov produkovaných v danom výrobnom procese. V poslednej fáze výroby sa využíva ku kontrole identifikačných znakov výrobkov ako sú štítky alebo podtlače obsahujúce informáciu o type výrobku alebo dátume expirácie výrobku. Pomocou prečítania textu strojovým videním sa dá eliminovať expedícia tovaru s uplynulou expiračnou dobou či nalepenie iného štítku, než je predpísané. Čítanie textu strojovým videním znázorňuje obrázok 1.4, na ktorom je vľavo zistené správne výrobné číslo produktu a vpravo je zistené nesprávne výrobné číslo produktu so zameneným znakom 'G' za 'C'.

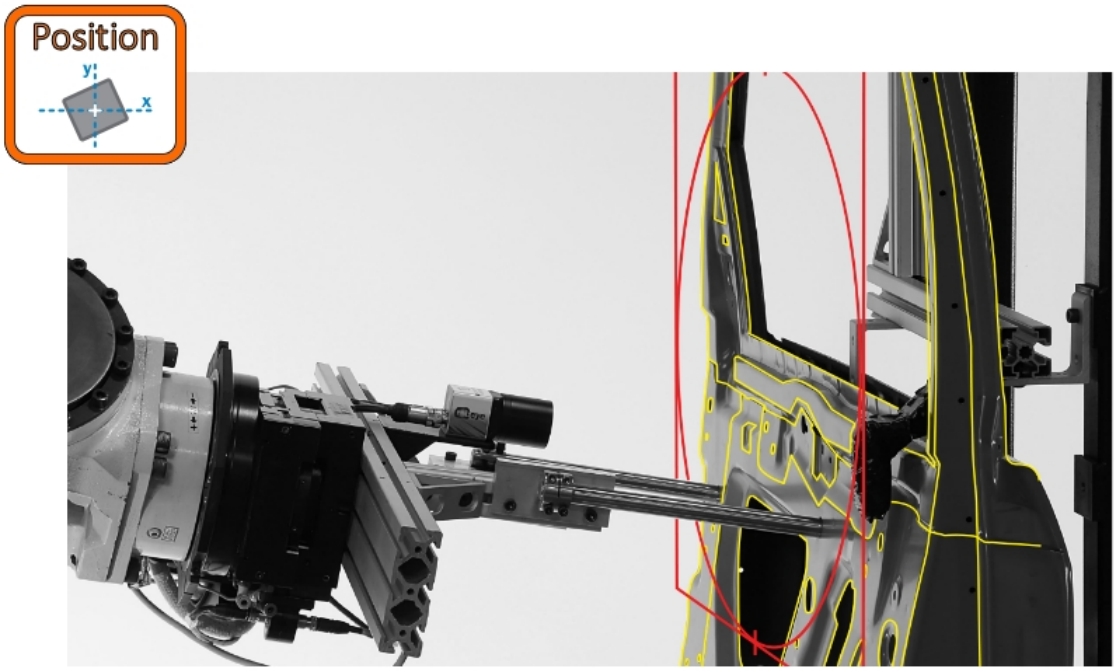


Obr. 1.4: Detekcia zámeny písmen [4]

- **Čítanie a verifikácia kódov** - čítanie kódov je typické pre prvú a druhú fázu výrobnej operácie, kedy sa zisťuje, či je k operácii pripravený správny diel

na správnom mieste. Detekované kódy sa delia na čiarové a maticové. Niekedy býva vytvorený nový kód počas priebehu výroby, ktorý sa verifikuje v poslednej fáze operácie. Čítanie alebo verifikácia kódov sú málokedy samostatnou úlohou strojového videnia vo výrobnej operácii. Čítanie a verifikácia kódov v automobilovom a farmaceutickom priemysle patrí k najrozšírenejšiemu spôsobu značenia maticovým kódom. Dôvodom je veľké množstvo informácie, ktoré je možné do značky s maticovým kódom umiestniť, taktiež spoľahlivosť značenia a redundancia kódu, ktorá zvyšuje spoľahlivosť čítania a umožňuje prečítať aj obsah poškodenej značky.

- **Identifikácia farieb** - úloha identifikácie farieb sa spravidla uplatňuje pri vstupnej alebo výstupnej kontrole, kde je potrebné rozpoznávať farby, prípadne sa môže jednať o jeden zo znakov pri identifikácii typu materiálu. V potravinárstve môže farba potraviny signalizovať zníženú kvalitu produktu. Ďalej inšpekciu pomocou identifikácie farieb možno používať na kontrolu, či bola pre tlač alebo náter správne aplikovaná farba.
- **Kontrola zostavenia a montáže** - pred montážou je nutné skontrolovať každý zmontovaný diel, či sú všetky jeho časti na mieste a v správnej polohe. Častou úlohou inšpekcie môže byť napr. kontrola zalisovania vodiča do konektoru, alebo kontrola osadenia poistkovej skrine správnymi poistkami. Na obrázku 1.5 je znázornená detekcia aktuálnej polohy automobilových dverí pri ich manipulácii robotickým manipulátorom.

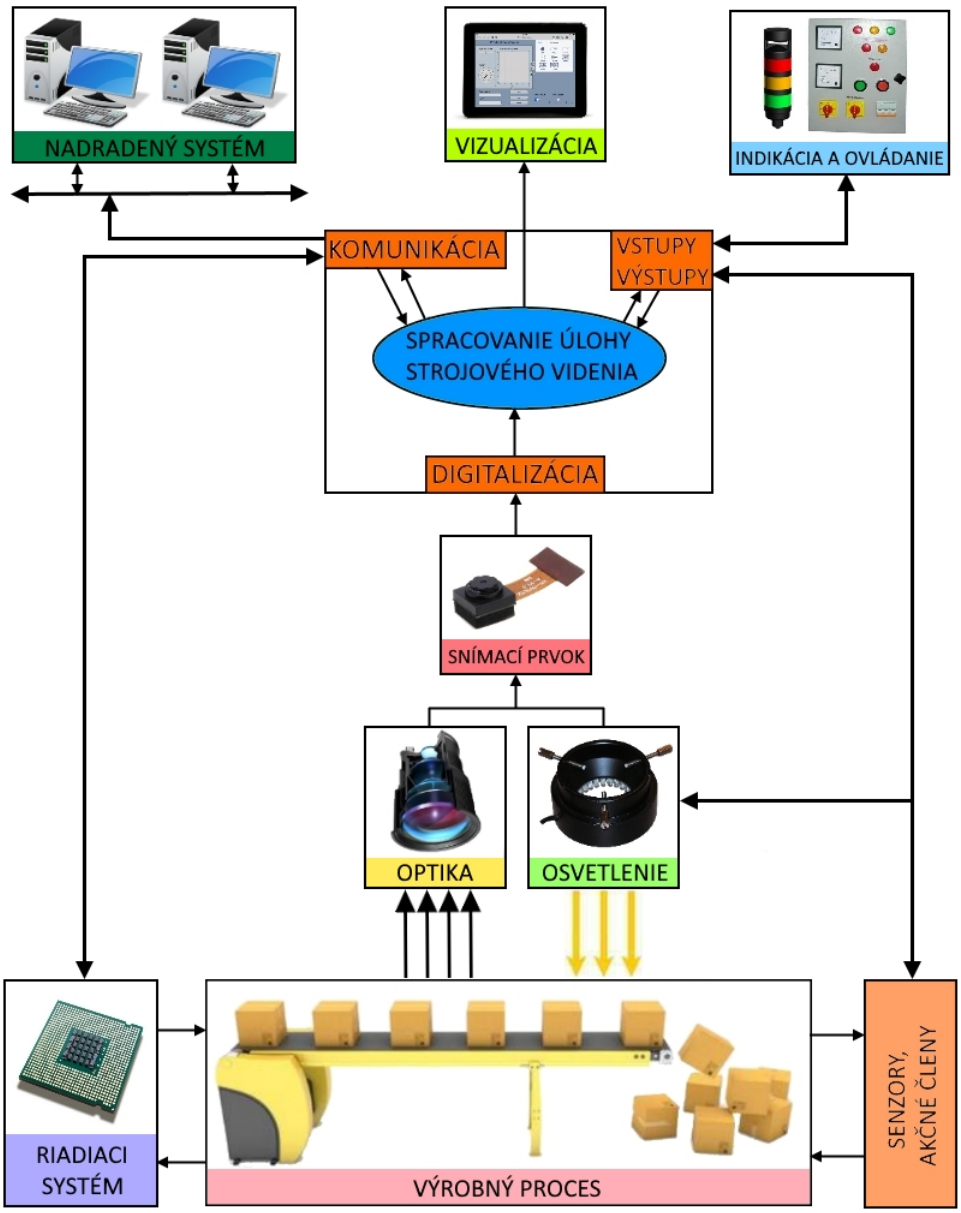


Obr. 1.5: Detekcia polohy pri montáži [4]

Uplatnenie nájdeme vo všetkých odvetviach priemyslu všade tam, kde je potrebné udržiavať určitý štandard kvality. Typickými príkladmi sú automobilový, elektrotechnický, potravinársky, tlačiarensky, baliaci a farmaceutický priemysel. [5][6]

1.3 Usporiadanie reťazca strojového videnia

V nasledujúcej kapitole sa budem zaoberať výhradne 2D rozpoznávaním obrazu, ktoré patrí zároveň k najrozšířenejším metódam strojového videnia. U tejto metódy získavania dát je nutná iba jedna kamera. Obrázok 1.6 popisuje blokovú schému, v ktorej je daný snímaný objekt (VÝROBNÝ PROCES) osvetlený zdrojom svetla alebo iným žiarením (OSVETLENIE) pre zvýraznenie charakteristických znakov snímaného objektu pre účely rozpoznávania. Na snímacom prvku kamery (SNÍMACÍ PRVOK), ktorého obraz je upravený optikou (OPTIKA) sa vytvorí dvojrozmerný jasový obraz objektu. Mieronosnou veličinou kamery s polovodičovým snímacím prvkom je náboj získavaný svetlocitlivými elementami obrazového snímača. Pre efektívne vyhodnotenie obrazovej informácie využívame výpočtovú techniku, z čoho vyplýva, že je nutné previesť analógovú mieronosnú veličinu do digitálneho tvaru. Touto operáciou je digitalizácia obrazu (DIGITALIZÁCIA). Informácia v digitálnej podobe je v počítači spracovaná a upravená vhodnými algoritmami, aby bola získaná požadovaná informácia o snímanom objekte (SPRACOVANIE ÚLOHY). Získaná informácia má taktiež digitálnu podobu a je distribuovaná do okolia pomocou digitálnych výstupov (VÝSTUPY) alebo prostredníctvom digitálneho komunikačného rozhrania (KOMUNIKÁCIA). Komunikačné rozhranie slúži vo veľa prípadoch aj k pripojeniu MMI zariadenia, pomocou ktorého je možné systém nastavovať a programovať (NADRADENÝ SYSTÉM). Pre zobrazenie výsledkov vyhodnotenia obrazu pre užívateľa sa používa vizualizačná aplikácia, ktorá býva spravidla zobrazená na zobrazovači (VIZUALIZÁCIA). Pre strojové videnie je charakteristické, že sa výmena získanej informácie s okolím realizuje ako spätná väzba na výrobný proces. Systém získava na jednej strane z procesu potrebné vstupné údaje (SENZORY). Typicky to býva povel k získaniu snímku v okamihu, keď je sledovaný objekt vo vhodnej polohe. Na druhej strane systém v závislosti na výsledku vyhodnoteného obrazu vykonáva akčný zásah v podobe indikácie (INDIKÁCIA A OVLÁDANIE), alebo vyradenia vadného kusu (AKČNÉ ČLENY). Zásah môžu zabezpečiť priamo digitálne výstupy systému strojového videnia alebo môže byť informácia o výsledku vyhodnotenia obrazu predaná do riadiaceho systému (RIADIACI SYSTÉM). [3][5]



Obr. 1.6: Obecné usporiadanie systému strojového videnia [3]

1.3.1 Snímací prvok

Prvým a zároveň najdôležitejším kľúčovým prvkom v reťazci hardwaru kamerového systému je snímací prvok, ktorý je akýmsi "okom" systému. Úlohou snímaču je zachytiť reálny obraz prostredia tvorený svetelným spektrom vo forme fotónov. Z úlohy snímania okolia snímačom vyplýva fyzikálny princíp, na základe ktorého snímač pracuje, jedná sa o vnútorný fotoelektrický jav. V dnešnej dobe sa v kamerách využívajú snímače hlavne na tomto princípe. K prudkému nárastu ich využitia došlo vďaka rozvoju polovodičov, pričom vznikli dva základné typy snímačov pre priemyselné aplikácie. Ako prvý vznikol snímač typu CCD, za ním nasledoval snímač CMOS,

ktorý je hlavne v posledných rokoch viac presadzovaný. [20]

CCD snímače

Prvý typ snímačov (obrázok 1.7 vľavo) pracuje na princípe analógového posuvného registra s využitím už spomínaného vnútorného fotoelektrického javu, pri ktorom po náraze fotónu do atómu je uvoľnený elektrón z valenčnej vrstvy do vodivostného pásma a prejde do excitovaného stavu, čoho výsledkom je vznik elektrického náboja. Každý snímač pozostáva zo samostatných miniatúrnych polovodičových segmentov zaznamenávajúcich svetlo, ktoré sa nazývajú pixely. V lineárnych snímačoch sa používa štruktúra pixelov vytvárajúcich vertikálny riadok, plošné snímače sa skladajú z viacerých takýchto stĺpcov pixelov, kde medzi každým svetlocitlivým stĺpcom je naviac ešte vertikálny posuvný register. Ten slúži k presunu náboja v snímači. Po expozícii snímača sa nahromadené náboje presunú do vertikálneho registra, prostredníctvom neho sa dostanú do horizontálneho registra, pomocou ktorého prejdú až k výstupu k obrazovému zosilňovaču a A/D prevodníku. Výstup snímača tohto typu prenosu náboja v plošnom snímači sa nazýva medziriadkový prenos (angl. interline transfer). Tento typ prenosu je v strojovom videní používaný najviac. Jeho nevýhodou však je, že takmer polovica plochy snímača je prekrytá registrami, ktoré nie sú citlivé na dopadajúce žiarenie. Táto nevýhodná vlastnosť môže byť čiastočne kompenzovaná mikrošošovkami. Existujú ešte ďalšie typy prenosu náboja na výstup v plošnom CCD snímači, napr. snímkový prenos (angl. frame transfer) alebo plne snímkový prenos (angl. full frame). [23] [21]

Výhodou CCD snímačov je vysoká miera, ktorou sa približuje k skutočnému snímanému farebnému obrazu, vysoká citlivosť, vysoký dynamický rozsah, malý šum, malé rozmery, vysoká geometrická presnosť pixelov a odolnosť voči vibráciám a elektromagnetickému poľu. [23]

Nevýhodou CCD snímačov je zvýšený odber energie, ďalej môže nastať efekt presvetlenia (angl. blooming), ktorý vzniká pri nadmernom ožiarení pixelu, pričom vznikne nadmerne veľký náboj, ktorý pretečie do okolitých pixelov. Ďalšou nevýhodou je nízka rýchlosť snímača, spôsobená sekvenčným čítaním snímača. [23]

CMOS snímače

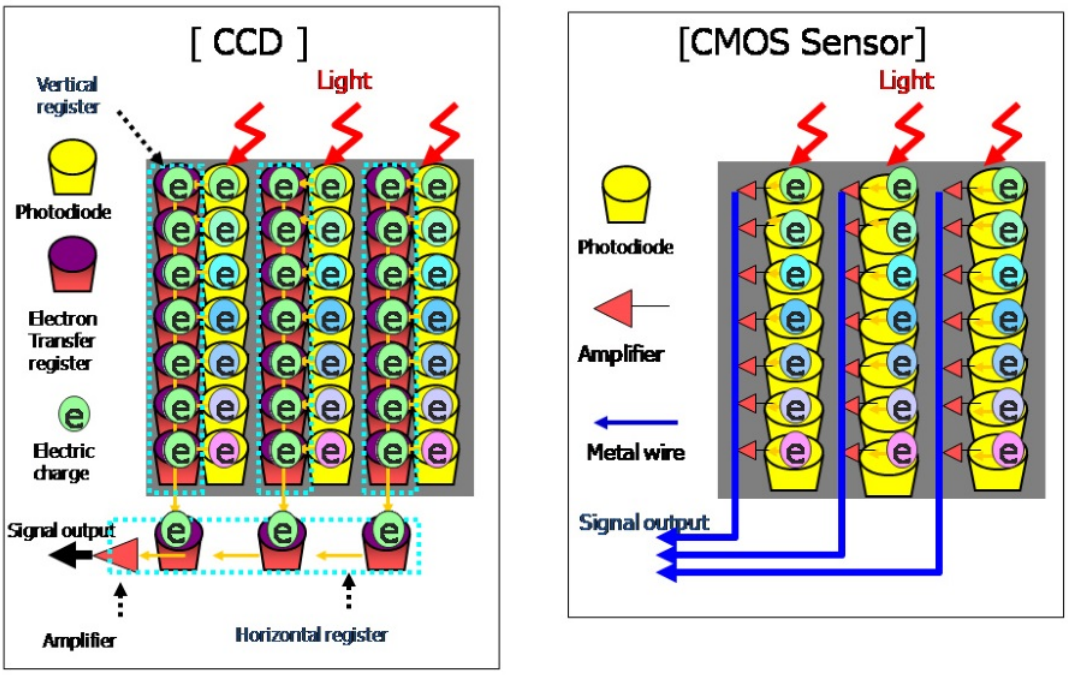
Druhý typ snímačov (obrázok 1.7 vpravo) pracuje rovnako na vnútornom fotoelektrickom jave. Zatiaľ čo CCD snímače posúvajú náboj z pixelov do jedného alebo viacerých výstupných zosilňovačov, CMOS snímače majú zosilňovač v každom pi-

xeli. Táto vlastnosť je najväčším rozdielom princípu činností popísaných dvoch typov snímačov. Tieto snímače so zosilňovačom v každom pixeli bývajú označované aj APS (angl. active pixel sensor). Technológia CMOS nie je novinka, avšak využívanie tohto typu snímača začalo len prednedávnom. V niektorých parametroch majú dokonca CMOS snímače navrch nad CCD. Pri CMOS snímačoch je potrebné zabezpečiť kontrolovanie začiatku a konca expozície, čiže expozičného času. Za týmto účelom využívame dve metódy. Prvá metóda je tzv. rolovacia uzávierka, kedy expozícia snímača začne s vyprázdnením nábojov, čo sa pri rolovacej uzávierke nevykonáva súčasne na celej ploche snímača, ale expozícia prebieha rad po rade. Mechanizmus metódy je lacný, ale spôsobuje skreslenie pohybujúcich sa objektov, ktoré snímač sníma. Tento typ uzávierky sa používa u bežných mobilných telefónov. Druhá metóda je tzv. globálna uzávierka, kde je celý snímač vystavený svetlu v jednom okamžiku, pričom sú obrazové dáta prenášané z celej plochy snímača v podstate súčasne. V kamerových systémoch strojového videnia sa využíva hlavne tento typ uzávierky. [23][22]

Hlavnou výhodou CMOS snímačov je vďaka zosilňovačom v každom pixeli možnosť jeho adresovania a čítania pomocou súradníc X a Y, vďaka čomu sa zrýchľuje čítanie snímky a znižuje spotrebu energie a produkciu zostatkového tepla. Snímače tohto typu sú menej finančne náročné na výrobu, vyžadujú len jedno prevádzkové napätie a možnosť zakomponovania elektronických prvkov na snímač. [23]

Nevýhodou CMOS snímačov je absencia uniformity zosilňovačov spôsobená výrobnými toleranciami. Nevýhodou oproti CCD snímačom je taktiež vyšší šum. [23]

Pri výbere vhodnej kamery, ktorá obsahuje aj snímací prvok, ktorý je teraz naším predmetom rozboru, je potrebné prihliadať hlavne na nasledujúce parametre:



Obr. 1.7: Porovnanie princípu činnosti technológie CCD a CMOS [24]

- **Rozlíšenie snímača** označuje celkový počet pixelov ako súčin vertikálnych a horizontálnych pixelov v snímači. Získaný súčin sa udáva v megapixeloch (Mpx). Rozlíšenia snímačov bývajú najčastejšie v rozmedzí od 640 x 480 pixelov (0,3 Mpx) do 2448 x 2050 pixelov (5 Mpx) v pomeroch obrazu 4:3, využívajú sa však aj iné veľkostí pomerov. [25]
- **Veľkosť snímača** je jedným z najdôležitejších parametrov, ktorý sa nepriamo podieľa na celkovej kvalite snímku a množstva prítomného šumu. Tento podiel na kvalite a množstve šumu vychádza z predpokladu, že čím je väčší snímač, tým sa doň zmestí viac pixelov alebo pixely väčších rozmerov. Z historických dôvodov sa veľkosť snímača označuje v zlomkoch palca, pričom skutočný rozmer snímača s hodnotou v palcoch nekorešponduje. [25][26]
- **Veľkosť pixela** je veľkosť svetlocitlivej plochy, pričom veľkosť tejto plochy resp. pixelu rozhoduje o tom, aký veľký náboj po dopade fotónov dokáže uložiť, čoho dôsledkom je väčší dynamický rozsah, teda citlivosť. Bežne v systémoch strojového videnia sú zaužívané veľkosti pixelov od 4 do 7 µm, výnimkou sú farebné CMOS snímače, u ktorých je veľkosť okolo 2 µm. [25]
- **Priestorové rozlíšenie** popisuje množstvo aktívnych pixelov v snímači. Pre správne určenie veľkosti rozlíšenia v aplikácii strojového videnia, je potrebné vedieť, aké najmenšie prvky majú byť detegované. [25]

1.3.2 Objektív

Druhý kľúčový prvok v hardware kamerového systému je optika. Tá slúži k usmerneniu lúčov svetla od snímanej oblasti do snímacieho prvku. Návrh príslušnej optiky pre kamerový systém realizujeme na základe nasledujúcich parametrov:

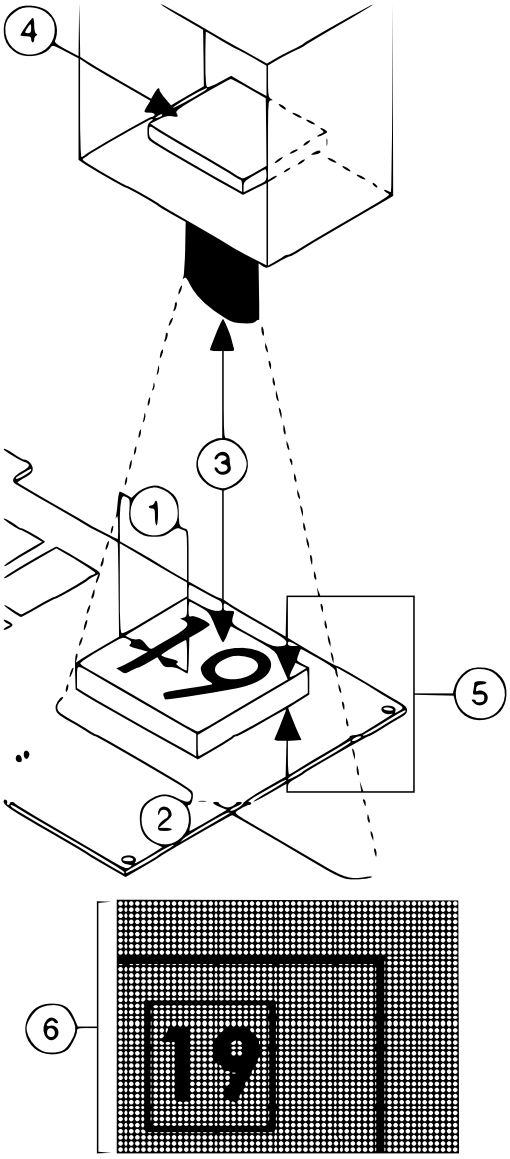
- **Rozlíšenie optiky (1)** je parameter poukazujúci na najmenšiu presnosť s akou je obraz snímaný. Pri návrhu platí, že rozlíšenie musí byť dvakrát väčšie ako je požadovaná presnosť podľa Nyquistovho kritéria. Ak by sme chceli zvýšiť presnosť snímania pri zachovaní snímacieho prvku, znižime tým veľkosť zorného poľa (2).
- **Zorné pole (2)** *(angl. Field of view)* je časť snímaného priestoru (6), ktorý je zachytávaný snímacím prvkom. Veľkosť tejto časti resp. uhlu je určený ohniskovou vzdialenosťou optiky a rozmerom snímacieho prvku (4). Pri voľbe optiky robíme výber hlavne na základe požadovanej veľkosti zorného poľa.
- **Pracovná vzdialenosť (3)** je vlastne vzdialenosť snímacieho prvku od snímanej plochy, bez započítania rozmeru objektívu. Jej veľkosť sa určuje na základe požadovanej veľkosti zorného poľa a ohniskovej vzdialenosti objektívu.
- **Hĺbka ostrosti (5)** popisuje hranicu, v ktorej je obraz ešte ostrý. Pri detekcii, napr. nápisov alebo plochých objektov, sa volí objektív s malou hĺbkou ostrosti, pri objemnom objekte, naopak s veľkou hĺbkou ostrosti.
- **Ohnisková vzdialenosť objektívu** je kolmá vzdialenosť prvej šošovky alebo člena optickej sústavy od matnice. Veľkosť ohniskovej vzdialenosti má vplyv na veľkosť obrazu premietaného na matnicu. Čím je ohnisková vzdialenosť objektívu kratšia, tým väčšiu časť scény môže zachytiť. Počíta sa podľa vzťahu:

$$f = \frac{y' * l}{y + y'} \tag{1.1}$$

  - f ... ohnisková vzdialenosť objektívu [mm]
  - l ... pracovná vzdialenosť (3) [mm]
  - y ... veľkosť snímanej oblasti (6) [mm]
  - y' ... veľkosť snímacieho senzoru (4) [mm]

- **Nastaviteľnosť ohniskovej vzdialenosti objektívu** existuje v troch variantoch, s pevnou ohniskovou vzdialenosťou, s premennou ohniskovou vzdialenosťou a so zoomom. Tretí variant sa líši od druhého v rozsahu nastaviteľnosti od 6 až do 48 mm bez ovplyvnenia ostrosti obrazu, pričom pri druhom je to iba spravidla rozsah od 3,5 do 8 mm s nutným zaostrením obrazu po každej zmene ohniskovej vzdialenosti.

- **Spôsob uchytenia objektívu** popisuje typ závitu, ktorý je k tomuto účelu použitý. Prvý typ je C-mount, pri ktorom je vzdialenosť fokusačného bodu šošovky od snímacieho prvku (ohnisková vzdialenosť) 17,52 mm. Druhý typ je CS-mount, pri ktorom je fokusačný bod vzdialený 12,52 mm. V súčasnej dobe je tento typ používaný najčastejšie. Pre rozmerovo veľké senzory sa používa F-mount, naopak pre veľmi malé S-mount. Pre komerčné fotoaparáty a kamery zas K-mount. [1][25]



Obr. 1.8: Popis parametrov objektívu [1]

1.3.3 Osvetlovač

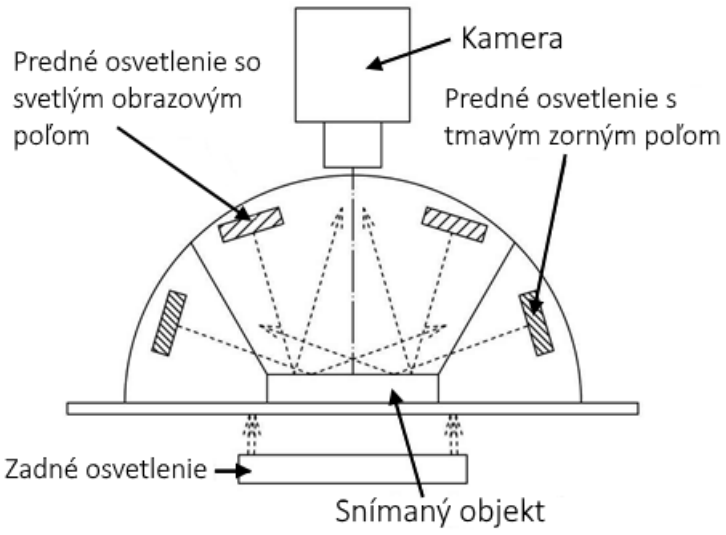
Pre vytvorenie spoľahlivej inšpekcie je dôležitý správny osvetľovač alebo ich kombinácia, ktorá vytvorí vhodné optické podmienky. Pri ďalšej práci so získanými snímkami, na ktorých je snímaný objekt vhodne osvetlený so zvýraznenými záujmovými bodmi, je potom možné sa vyhnúť rôznym zložitým operáciám predspracovania obrazu. Na základe znalostí geometrie svetla, typov osvetlenia a optických vlastností skúmaných vzoriek, možno navrhnúť efektívny zdroj svetla, ktorý bude schopný splniť tri kľúčové vlastnosti:

- **Maximalizovanie kontrastu záujmových bodov objektu**
- **Minimalizovanie kontrastu častí, ktoré nie sú predmetom inšpekcie**
- **Odolnosť voči rušivým vplyvom**

Pred samotným návrhom osvetľovacej sústavy, ktorá dodrží tri spomínané vlastnosti, je potrebné zistiť, aké vplyvy môže mať **geometria osvetlenia**, **vlnová dĺžka svetla** alebo použitý **zdroj svetla** na celý systém.

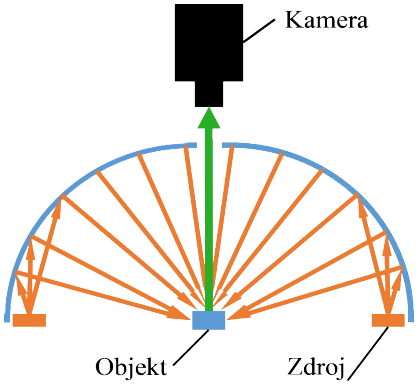
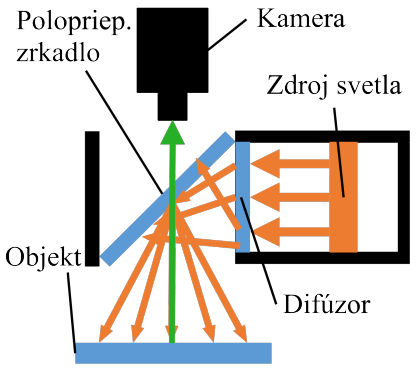
Geometria osvetlenia

Obecné rozloženie geometrických polí osvetlenia je vidieť na obrázku 1.9. [8]

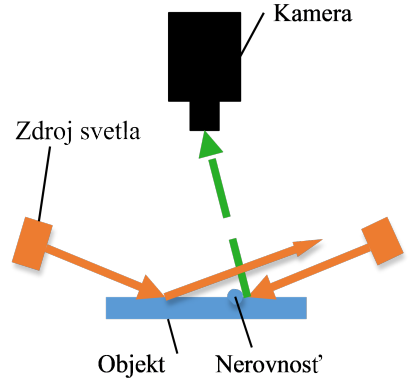


Obr. 1.9: Geometrické rozloženie polí osvetlenia

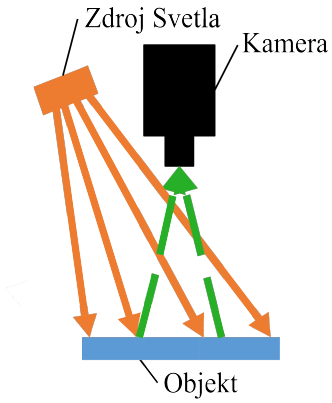
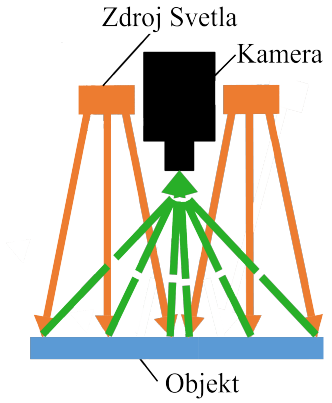
Na spomínaný obrázok nadväzujú tabuľky nižšie, ktoré popisujú konkrétne druhy osvetľovačov patriacich do príslušnej skupiny geometrie osvetlenia:

Predné osvetlenie so svetlým difúznym obrazovým poľom		
Kupolovitý osvetlovač		
	Výhody	• Poskytuje rozptýlené svetlo • Okolo objektu a na objekte nevzniká tieň • Eliminuje odlesky • Vhodné pre čítanie znakov, zaoblené alebo lesklé povrchy
	Nevýhody	• Veľké rozmery osvetlovača • Požiadavka na minimálnu vzdialenosť od ožarovanej plochy • Nutnosť pokryť celý sledovaný objekt
Osvetlovač s axiálnym osvetlovacím poľom		
	Výhody	• Dokonalý spôsob osvetlenia s rozptýleným svetlom a jasným obrazovým poľom • Svetlo dopadá rovnomerne z celej plochy osvetlovača na celý objekt • Eliminuje tiene • Vhodné pre čítanie znakov, zaoblené alebo lesklé povrchy
	Nevýhody	• Vysoká cena • Obmedzené zorné pole • Náročná implementácia

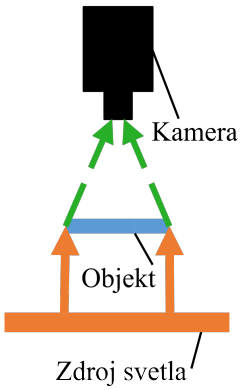
Tab. 1.1: Osvetlovače s predným osvetlením so svetlým difúznym obrazovým poľom

Predné osvetlenie s tmavým zorným poľom		
Osvetľovač s temným zorným poľom		
	Výhody	• Zdroj svetla je takmer kolmý k osi objektívu • Vhodné pre inšpekciu povrchových nerovností • Vysoký kontrast obrazu
	Nevýhody	• Vyžaduje malú vzdialenosť od ožarovaného povrchu objektu • Nevhodné pre osvetľovanie hladkých povrchov • Iba úzky okruh vhodných aplikácií

Tab. 1.2: Osvetľovač s predným osvetlením s tmavým zorným poľom

Predné osvetlenie so svetlým smerovým obrazovým poľom		
Osvetľovač s plošným osvetľovacím poľom		
	Výhody	• Jednoducho nastaviteľný smer osvetlenia • Rozptýlené svetlo, vhodné pre vytváranie kontrastu a zvýraznenie detailov objektu
	Nevýhody	• Len pre nenáročné aplikácie • Môže spôsobovať nežiaduce odrazy alebo tiene • Osvetlenie je nerovnomerné
Kruhový osvetľovač		
	Výhody	• Lúče svetla idú koaxiálne s objektívom, čo eliminuje tiene • Nízke obstarávacie náklady • Jednoduchá montáž priamo na objektív kamery
	Nevýhody	• Nižší osvit objektu oproti ostatným osvetľovačom

Tab. 1.3: Osvetlovače s predným osvetlením so svetlým smerovým obrazovým poľom

Zadné osvetlenie		
Osvetľovač so zadným svetlom		
	Výhody	• Vhodné pre získavanie obrysu skúmaného objektu • Umožňuje získavať obraz objektu v priehľadnom puzdre, ktoré by inak spôsobovalo odrazy • Nízke obstarávacie náklady
	Nevýhody	• Potrebný priestor pre osvetľovač pod pozorovaným objektom • Zobrazuje iba hrany objektu

Tab. 1.4: Osvetľovač so zadným osvetlením [9][10]

2 SPRACOVANIE OBRAZU

Pred samotnou detekciou znakov v obraze je nutné najskôr detegovať objekt, na ktorom sa znaky nachádzajú. Zo znalosti umiestnenia objektu v obraze sme potom schopní sa zamerať na oblasť obrazu s vopred známym umiestnením znakov na objekte. Aby bolo možné detegovať objekt a určiť jeho hranicu, treba obraz predspracovať *(angl. preprocessing)*, pričom výber vhodného filtra závisí od konkrétnej aplikácie. Pre zložitejšie scény je potrebné aplikovať aj algoritmy potlačenia šumu a transformácie invariantné voči natočeniu a zmene mierky. Ak je objekt detegovaný a lokalizovaný v obraze, je možné opísať jeho hranice pomocou algoritmov detekcie hrán. [11]

Pre detekciu objektov a znakov v obraze existuje množstvo techník, v nasledujúcich podkapitolách popisujem iba tie techniky, ktoré využívam vo vlastnej aplikácii rozpoznávania.

2.1 Filtrovanie obrazu

Filtráciu obrazu používame hlavne vtedy, keď potrebujeme z obrazu odstrániť šum, vyhladiť obraz, rozmazať hrany alebo ak chceme získať hranový obraz. Medzi najčastejšie filtre patria:

- **Spriemerovanie**
- **Gaussov filter**
- **Mediánový filter**
- **Filtrácia pomocou rotujúcej masky**
- **Laplaceov filter**
- **Sobelov filter**
- **Cannyho hranový detektor**

Ďalej sa budem detailnejšie zaoberať iba spriemerovaním, mediánovým filtrom a Cannyho hranovým detektorom, ktoré sú použité vo vlastnej aplikácii. Všeobecne pri filtrovaní sa používa konvolučná maska alebo inak konvolučné jadro (convolutional kernel), bežne 3x3 alebo 5x5, pomocou ktorej je určená vždy nová hodnota každého pixelu obrazu. [13]

2.1.1 Spriemerovanie

Patrí medzi najjednoduchšie spôsoby vyhladzovania šumu v obraze, pričom dochádza k rozostreniu obrazu. Každý pixel obrazu sa nahradzuje aritmetickým priemerom jasových hodnôt pixelu susedných pixelov. Filtrácia obyčajným priemerovaním je v podstate špeciálnym prípadom konvolúcie, kde má konvolučná maska tvar (3x3): [16]

$$h = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{2.1}$$

2.1.2 Mediánový filter

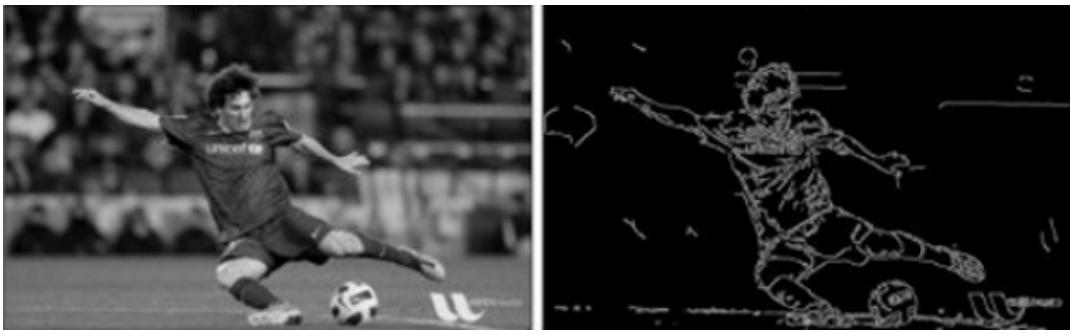
Mediánový filter je zložený z posuvného okna, ktoré sa skladá z nepárneho počtu pixelov. Medián ξ je vybraný prvok zo súboru usporiadaných jasových hodnôt pixelov, v ktorom vždy pri nepárnom počte prvkov je to prvok:

$$\xi = \frac{p_{n+1}}{2} \tag{2.2}$$

teda prostredná hodnota usporiadaných prvkov vzostupne. Pokiaľ by sa pracovalo s párnym počtom prvkov, vyberú se dva prostredné prvky, z ktorých sa spočíta ich aritmetický priemer. Princíp filtrácie spočíva v posuve masky po obraze a výbere mediánu z hodnôt ležiacich pod touto maskou. Tento druh filtrácie je výborným nástrojom pre elimináciu impulzového šumu alebo šumu typu soľ a korenie z obrazu. [13][17]

2.1.3 Cannyho hranový detektor

Jedná sa o detektor, ktorý podstatne zvýrazňuje hrany objektu. Algoritmus detektoru pozostáva zo štyroch krokov, v prvom je vo vstupnom obraze eliminovaný šum aplikovaním Gaussovho filtra. V druhom sa určuje veľkosť a smer gradientu pomocou Sobelovho operátora. V treťom kroku sú eliminované body resp. pixely, ktoré nie sú lokálnym minimom. Čiže sú nájdené pixely, ktoré majú vo svojom malom okolí maximálny gradient, čo znamená, že susedné pixely v smere gradientu majú menší gradient. Takéto pixely v obraze sú považované za miesta, ktorými určite prechádza hrana. V poslednom kroku sa aplikujú dva prahy T1 a T0, ktoré slúžia ako hranice hysterézie, čiže ak gradient pixelu prevyšuje horný prah, pixel je hranovaný a ak je nižší ako dolný prah, nedochádza u neho k hranovaniu. [15]
Na obrázku 2.1 je vľavo ukázaný pôvodný šedotónový obrázok a vpravo obrázok s aplikovaným Cannyho hranovým detektorom.



Obr. 2.1: Porovnanie obrázku pred aplikovaním Cannyho hranového detektoru a po aplikovaní [15]

2.2 Morfologické operácie

Predstavujú relatívne samostatnú etapu spracovania obrazu vďaka odlišnému matematickému aparátu aplikovaného pri výpočtoch. Aplikujú sa na binárne obrazy, ktoré sú vnímané ako množiny. Jednotlivé metódy sú preto definované pomocou množinových operátorov. Morfologické operácie zabezpečujú odstránenie šumu, zjednodušenie tvaru, stenšenie alebo zosilnenie kontúr alebo popisujú tvarové vlastnosti kontúr číselnými príznakmi, ako sú napr. plocha, obvod, dĺžka a iné. Medzi najpoužívanejšie operácie patrí dilatácia, erózia, otváranie a uzatváranie. Vo vlastnej aplikácii rozpoznávania využívam dilatáciu, eróziu, uzatváranie a euklidovskú vzdialenosť, ktoré sú popísané v nasledujúcich podkapitolách. [16]

2.2.1 Dilatácia

Dilatácia *(angl. dilation)* po aplikovaní spôsobí expanziu objektov vo všetkých smeroch. Táto transformácia zmodifikuje všetky body pozadia susediace s objektmi na body objektov, čiže pridá objektom jednu vrstvu na úkor pozadia. Dochádza samozrejme ale aj ku zväčšeniu bodov predstavujúcich šum. Dilatáciu môžeme definovať aj ako zjednotenie posunutých bodových množín vzťahom: [16]

$$X \oplus B = \bigcup_{b \in B} X_b \tag{2.3}$$

Na obrázku 2.2 je vľavo ukázaný pôvodný prahovaný obrázok a vpravo obrázok s aplikovanou dilatáciou.

Obr. 2.2: Porovnanie obrázku pred aplikovaním dilatácie a po aplikovaní [12]

2.2.2 Erózia

Erózia *(angl. erosion)* po aplikovaní zmení všetky body objektov susediace s pozadím na body pozadia, čiže odoberie objektom jednu vrstvu bodov na úkor pozadia. Dochádza tým pádom k odstráneniu objektov o veľkosti 1 pixelu, ako je napr. šum na pozadí. Touto operáciou sme schopní taktiež oddeliť dotýkajúce sa objekty alebo získať obrys objektu po odčítaní erodovaného objektu od pôvodného. Eróziu môžeme definovať aj ako prienik súboru posunutých bodových množín podľa vzťahu: [16]

$$X \Theta B = \bigcap_{b \in B} X_{-b} \tag{2.4}$$

Na obrázku 2.3 je vľavo ukázaný pôvodný prahovaný obrázok a vpravo obrázok s aplikovanou eróziou.

Obr. 2.3: Porovnanie obrázku pred aplikovaním erózie a po aplikovaní [12]

2.2.3 Uzavretie

Operácia uzavretia *(angl. closing)* je vlastne dilatácia nasledovaná eróziou. Aplikovanie v obraze spojí oddelené objekty, ktoré sú blízko seba, zaplní malé diery a vyhladí obrys vyplnením úzkych zálivov. Uzavretie je popísané vzťahom: [16]

$$X \bullet B = (X \oplus B) \Theta B \tag{2.5}$$

Na obrázku 2.4 je vľavo ukázaný pôvodný prahovaný obrázok a vpravo obrázok s aplikovaným uzavretím.

Obr. 2.4: Porovnanie obrázku pred aplikovaním uzavretia a po aplikovaní [13]

2.2.4 Euklidovská vzdialenost'

Euklidovská vzdialenost' medzi bodmi p a q v obraze je definovaná vzt'ahom:

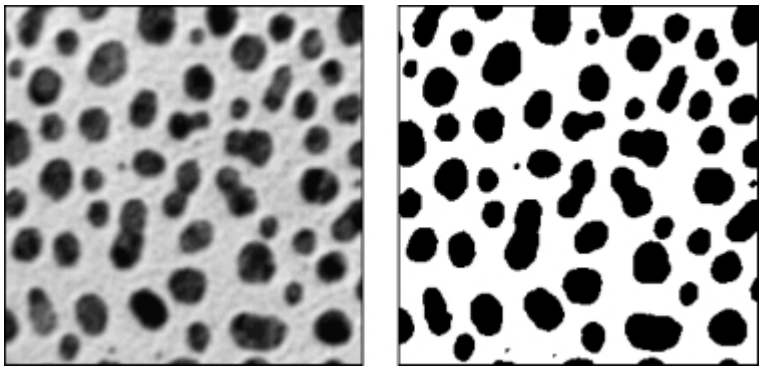
$$D_E(p,q) = \sqrt{(x1 - x2)^2 + (y1 - y2)^2} \tag{2.6}$$

2.3 Prahovanie obrazu

Prahovanie *(angl. thresholding)* je najjednoduchšia metóda segmentácie, ktorá je založená na hodnotení úrovne jasu každého pixelu. Pri prahovaní je hľadaná taká hodnota prahu v obraze, pri ktorej sú všetky hodnoty jasu nižšie ako prah ekvivalentný pozadiu, pričom všetky hodnoty vyššie ako prah sú ekvivalentné popredíu obrazu. Jedná sa o transformáciu, ktorá mapuje vstupný obraz *f(i,j)* na výstupný obraz *g(i,j)* podľa vzťahu:

$$g(i,j) = \begin{cases} 1 & \textit{pre } f(i,j) \geq u \\ 0 & \textit{pre } f(i,j) < u \end{cases} \tag{2.7}$$

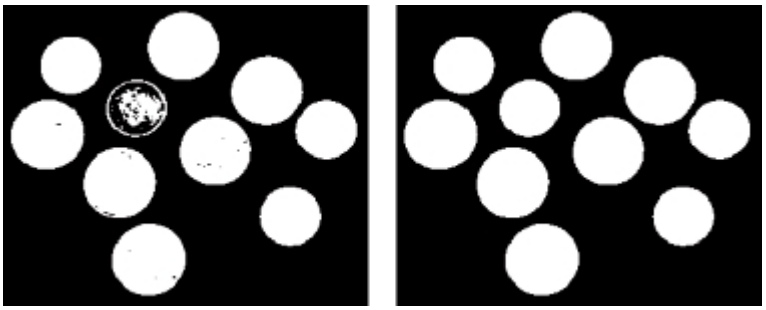
pričom u je prah. Hodnota prahu sa môže zadať ručne alebo automaticky. Ak chceme automaticky nastavovať prah, často volíme adaptívne prahovanie. Rozdiel medzi adaptívnym a jednoduchým prahovaním je, že pri adaptívnom sa prahová hodnota vypočítava zvlášť pre každý bod obrazu. [18]
Na obrázku 2.5 je vľavo ukázaný pôvodný šedotónový obrázok a vpravo obrázok s aplikovaným prahovaním.



Obr. 2.5: Porovnanie obrázku pred aplikovaním prahovania a po aplikovaní [16]

2.4 Záplavové vypĺňanie

Algoritmus záplavového vypĺňania *(angl. flood fill)* vypĺňa oblasti rastrového obrazu, pričom pre vypĺňanie hľadá susedné body rovnakej farby (alebo podobnej), ktoré vytvárajú hranicu alebo spojitú oblasť. Na začiatku je vybraný pixel obrazu, tzv. semienko, okolo ktorého sú následne zafarbené všetky susedné pixely, ktoré majú podobnú farbu. Nové zafarbené pixely sa stávajú semienkami. Zafarbovanie obrazu končí vo chvíli, keď už nie sú žiadne semienka k zafarbeniu. Výber susedných pixelov býva často realizovaný ako 4-okolie alebo 8-okolie. [18]
Na obrázku 2.6 je vľavo ukázaný pôvodný prahovaný obrázok a vpravo obrázok s aplikovaným záplavovým vypĺňaním.



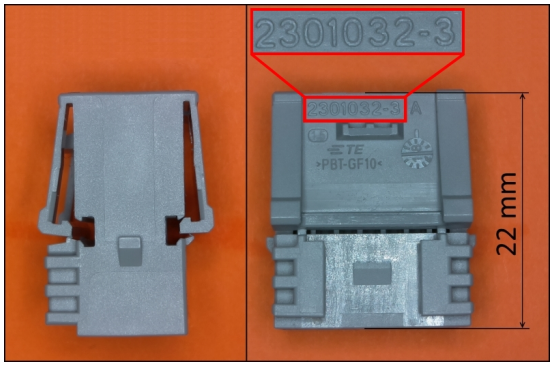
Obr. 2.6: Porovnanie obrázku pred aplikovaním zaplavenia a po aplikovaní [14]

3 POPIS SNÍMANÉHO OBJEKTU

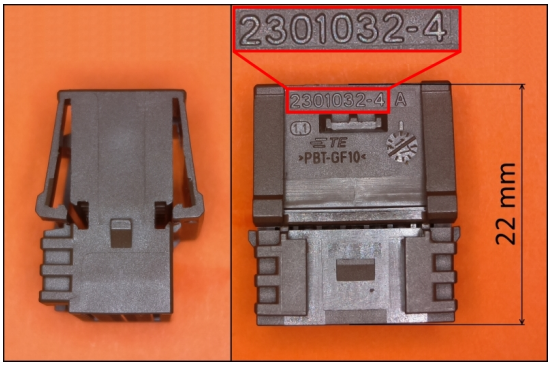
Spoločnosť *TE Connectivity* sa zameriava na široký okruh elektronických produktov, senzorových komponenty a snímačov, ktoré sú využívané v rôznych priemyselných odvetviach. Do popredia treba dať, že tieto produkty sú dokonalo odolné voči nepriaznivým podmienkam. Celkový počet pobočiek, ktoré slúžia svojím zákazníkom, je v 140 krajinách. Diplomová práca je venovaná jednej z pobočiek tejto spoločnosti. Práca je určená konkrétne pre závod *TE Kuřim* nachádzajúci sa 7 km od Brna. Pobočka je svojim výrobným programom zameraná na automobilový priemysel.
V diplomovej práci je primárnou úlohou z oblasti strojového videnia optické rozpoznávanie znakov, tzv. OCR. Jedná sa konkrétne o inšpekciu číslic na jednom konektore dvojakej farby a súčiastke, všetko od firmy *TE Connectivity*. Jednotlivé konektory a súčiastka sú vyobrazené v dvoch pohľadoch na obrázkoch nižšie. Zoskupenia čísel popisujú priradenie konektoru do príslušnej výrobnej šarže.

Prvý objekt - šedý konektor má číslo šarže vylisovaný do hĺbky, povrch číslic je lesklý, zvyšok povrchu konektoru, kde nie sú výlisky, je matný, ako je vidieť na obrázku 3.1 zvýraznené v červenom rámiku. Vhodným výberom osvetlenia sa zvýraznia v tomto prípade hrany číslic. Rozmer najdlhšej hrany konektoru je 22 mm. **Druhý objekt - hnedý konektor** je konštrukčne identický ako šedý konektor iba s rozdielom farby, ako je vidieť na obrázku 3.2. **Tretí objekt - čierna súčiastka** má číslo šarže vylisované na bočnej strane, ako je vidieť na obrázku 3.3 zvýraznené v červenom rámiku. Číslice sa nachádzajú v 1 mm hlbokej lesklej drážke a sú vypuklé. Vhodným výberom osvetlenia sa zvýraznia kontúry vypuklých číslic. Rozmer najdlhšej hrany súčiastky je 49 mm.

Počas výrobného procesu môžu nastať rôzne nepredvídateľné chyby nielen od vstrekovacieho lisu a vstrekovacej formy, v ktorých sa priamo konektory vyrábajú, ale aj od samotného zapracovávaného plastu a mnohých ďalších. To môže mať za následok deformáciu konektoru a s ním spojenú nečitateľnosť číslic na ňom, čo značí vadný kus. Ďalšou neprípustnou chybou pri inšpekcii je možná zámena číslic výrobnej šarže počas výrobného procesu, čo značí tiež chybný kus.



Obr. 3.1: Šedý konektor so zvýraznenými číslicami



Obr. 3.2: Hnedý konektor so zvýraznenými číslicami

Obr. 3.3: Čierna súčiastka so zvýraznenými číslicami

4 HARDWAROVÉ VYBAVENIE

4.1 Raspberry Pi

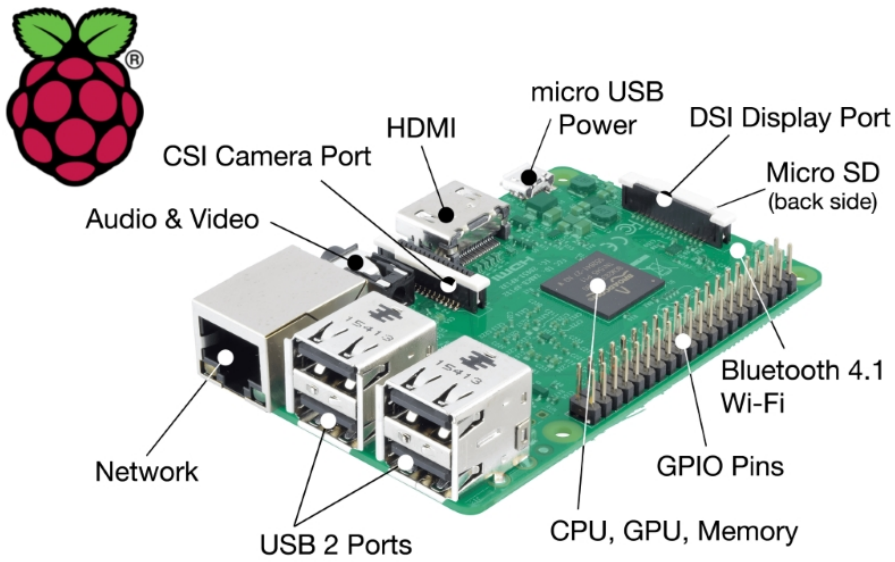
Raspberry Pi (ďalej len RPi) je priekopníkom mezi mikropočítačmi pre embeded systémy, v podstate je to vreckový počítač o veľkosti kreditnej karty (85,6 mm x 56,0 mm x 21,0 mm). Vyvíja ho britská nadácia *Raspberry Pi Foundantion* s cieľom podporiť výučbu základov programovania na školách. Vďaka svojej univerzálnosti sa rozšíril medzi širokú verejnosť rýchlym tempom. Užívatelia ho často využívajú pre domácu automatizáciu alebo aj v priemysle.
RPi sa dá použiť ako plnohodnotný počítač po pripojení vstupných zariadení, ktorými sú myš a klávesnica, klasicky prostredníctvom USB portu. Taktiež je možné pripojiť výstupné zariadenie, ktorým je napríklad display, a to prostredníctvom HDMI konektoru alebo kompozitným RCA konektorom. Pre prípadné pripojenie LCD panelu obsahuje RPi DSI konektor. Prednosťou Rpi je podstatne nižšia spotreba energie, ktorá sa pohybuje okolo 6,7 W (1,34 A) pri vyššom zaťažení procesoru. Napája sa za pomoci microUSB (rovnakým, aké majú súčasné smartfóny) alebo priamym pripojením 5V svorkami na zbernicu GPIO. Zbernica GPIO je bezpochyby jedna z periférií, ktorá robí RPi výnimočným oproti štandardným počítačom. Tá obsahuje okrem programovateľných vstupov a výstupov tiež zbernice UART, I2C a SPI. Ďalej RPi obsahuje ethernetový port a najnovšie modely aj integrovanú Wi-Fi anténu. Týmto spôsobom je možné sa spojiť s RPi cez lokálnu sieť a pripojiť ho na internet. Na doske RPi sa nachádza taktiež Audio výstup vo forme 3,5 mm jacku. Pripojenie kamery k RPi je možné prostredníctvom USB alebo CSI portu, toto rozhranie je aktuálne vyhradené iba pre RPi kamerový modul, iné využitie nemá. RPi v sebe nemá zabudovaný žiadny pevný disk alebo flash pamäť pre operačný systém a dáta. Namiesto toho disponuje slotom pre SD kartu (novšie modely pre microSD), z ktorej bootuje operačný systém a zároveň slúži aj ako pamäť programov a dát.
Prvá verzia RPi prišla na trh v roku 2012, vývoj na ňom pokračuje v dnešnej dobe a bude pokračovať aj v budúcnosti. Aktuálne je RPi dostupné na trhu vo forme viacerých modelov od počiatku jeho existencie a to vo verziach A, B, A+, B+, 2B, 2B+, 3B, 3A+, 3B+. Existujú aj RPi triedy Zero pre drobné aplikácie. Rozdiel medzi nimi je hlavne vo výkonoch procesorov, ale aj iných vlastností a je značný. [37]

Vo vlastnej aplikácii strojového videnia, resp. optickej inšpekcii je použitá takmer najvyššia trieda, konkrétne **Raspberry Pi 3 Model B** (obrázok 4.1), ktoré má vysoký a dostačujúci výpočtový výkon pre aplikáciu. RPi 3 má až tri možnosti ako sa pripojiť na sieť, resp. internet, a to pripojením klasického konektoru RJ45 do

ethernetu, ďalej kompatibilným Wi-Fi adaptérom do USB a nakoniec pomocou integrovanej Wi-Fi antény, ktorá sa nachádza na samotnej doske plošného spoja RPi 3. Vo vlastnej aplikácii sa využíva pripojenie pomocou integrovanej antény, využíva sa teda bezdrôtová komunikácia medzi nadradeným systémom a RPi. Pre zvýšenie výpočtového výkonu RPi používa pre napájanie oficiálny microUSB napájací zdroj, ktorý je schopný dodať veľkosť prúdu až do 2,5 A. Popis parametrov mikropočítača je v tabuľke 4.1:

Parametre	**Raspberry Pi 3 Model B**
Dátum vydania	29. Február 2016
SOC	Broadcom BCM2837
CPU	ARM Cortex-A53 64-bit
Počet jadier	4
Frekvencia CPU	1.2 GHz
GPU	Dual Core Video Core IV 1080p@30
RAM	1 GB DDR2
USB 2.0	4
Video výstup	HDMI, TRRS
Video vstup	15-pin MIPI camera interface (CSI), USB
Audio výstup	HDMI, 3,5 mm jack
Dátové úložisko	MicroSD
Ethernet	10/100 Mbit/s
Bezdrôtové pripojenie	Wi-Fi 802.11n, Bluetooth® 4.1 LE
Napájacie napätie	5V DC cez MicroUSB alebo GPIO
Prúdový odber	1.34 A
Rozmery	85,6 mm x 56,5 mm x 17 mm

Tab. 4.1: Parametre Raspberry Pi 3 Model B [33]



Obr. 4.1: Raspberry Pi model 3B s popisom periférií [32]

4.2 Kamerový modul

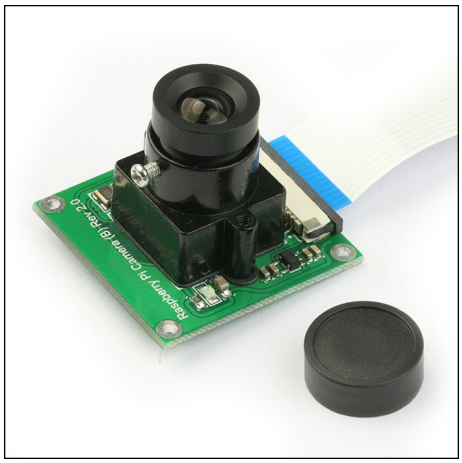
Jedným z najdôležitejších bodov práce bol výber vhodného kamerového modulu s optikou pre vlastný systém strojového videnia, pomocou ktorého je získavaný snímok objektu s výrobným číslom šarže. Pri výbere kamerového modulu bola podmienkou **kompatibilita** s mikropočítačom Raspberry Pi 3 Model B a **vysoká rýchlosť prenosu dát** z kamery do mikropočítača. Aktuálne existujú dve možnosti rozšírenia RPi o modul kamery. Prvou možnosťou je univerzálnejšie riešenie pomocou bežnej webkamery, ktorá sa pripojuje do USB portu. Druhou možnosťou je použiť špeciálny CSI modul určený výhradne pre mikropočítače typu RPi.

Na základe tabuľky 4.2 bol zvolený pre vlastnú aplikáciu strojového videnia ako vhodnejší monitorovací modul **Raspberry Pi kamera**. Rozhodujúcimi parametrami, na základe ktorých bola zvolená kamera, ako už bolo písané vyššie, je plná kompatibilita a znateľne vyššia prenosová rýchlosť. K tomu sa pridala ešte vyššia kvalita obrazu a nižšia cenová relácia.

Vlastnosti	**USB webkamera**	**Raspberry Pi kamera**
Pripojenie	USB port	CSI konektor
Nutnosť ovládačov	vo väčšine prípadov ÁNO, zlá dostupnosť ovládačov	NIE
Kompatibilita	ČIASTOČNÁ, kamery od rôznych výrobcov vykazujú chyby súvisiace s rozlíšením	ÁNO, plná
Maximálne overené rozlíšenie obrazu	2 Mpx (1920x1080) majú najkvalitnejšie kamery-stredne kvalitný obraz	5 Mpx (2592x1944) rev 1.3 8 Mpx (3264x2448) rev 2.1 pri oboch kvalitný obraz
Maximálna prenosová rýchlosť dát z kamery	240 Mb/s na jednu dátovú cestu (USB 2.0 rýchlosť Half-duplex)	1Gb/s na jednu dátovú cestu
Upevnenie objektívu	Už integrovaný objektív	M12 CS-mount
Cena	najkvalitnejšia rada Logitech Webcam C920 88 €	25 € rev 1.3 34 € rev 2.1

Tab. 4.2: Porovnanie vlastností kamerových zariadení [27][28]

Zvolený bol konkrétne kamerový modul **Raspberry Pi Camera (B)** (ďalej len RPiCam), ktorý je ukázaný na obrázku 4.2). Jedná sa o oficiálny produkt spoločnosti *Waveshare Electronics CO. Ltd.*, ktorá sa zaoberá výrobou a dodávaním špičkovej elektroniky špecializovanej na moduly, komponenty a vývojové platformy, ako aj doplnky pre RPi. Existuje celý rad modifikácií modulu RPiCam, od (B) až po (I), každá trieda má iné vlastnosti. Navrhnutý modul kamery disponuje malým, v priemere 14 mm veľkým objektívom, s mechanicky nastaviteľnou ohniskovou vzdialenosťou. [29]



Obr. 4.2: Raspberry Pi Camera (B) [29]

V nasledujúcej tabuľke 4.3 sú prehľadne popísané parametre kamerového modulu RPiCam:

Technológia senzoru	CMOS
Rozmer senzoru	1/4”
Ohnisková vzdialenosť objektívu	3.6 - 6.0 mm (nastaviteľná)
Zorný uhol objektívu (diagonálny)	75.7°
Numerická apertúra objektívu	2.0
Pripojovací závit objektívu	M12 x 0.5
Vonkajší priemer objektívu	14 mm

Tab. 4.3: Prehľad parametrov Raspberry Pi Camera (B)

RPiCam modul je zariadenie určené iba pre RPi. Je schopný vytvárať jednotlivé snímky alebo snímať video, pre túto činnosť ponúka Raspbian vstavané funkcie. RPiCam sa skladá z plošného spoja o rozmeroch 32 mm x 32 mm s CMOS senzorom OmniVision OV5647, ktorý je veľký 3,67 mm x 2,74 mm [31], a ohybného 15-žilového káblu dlhého 15 cm. Kábel je pripojený do CSI konektoru, ktorý sprostredkováva priame spojenie kamery s procesorom RPi. Rozhranie CSI poskytuje dátovú a riadiacu časť. Datová časť je iba jednosmerná z kamery do RPi s hodinovým signálom. Riadiaca časť je obojsmerná a kompatibilná s I2C rozhraním. Ďalej sa modul kamery skladá zo štandardnej 12 mm CS-mount (objímky), do ktorej sa skrutkuje objektív. Miera naskrutkovania resp. odskrutkovania mení ohniskovú vzdialenosť objektívu, pričom je možné objektív aretovať malou bočnou skrutkou.

Pre vlastnú aplikáciu strojového videnia je potrebné nastavenie ohniskovej vzdialenosti objektívu blížiacej sa ku hodnote:

$$f = \frac{y' * l}{y + y'} = \frac{3,67 * 100}{60 + 3,67} = \mathbf{5{,}76\ mm}, \tag{4.1}$$

ktorú objektív kamerového modulu zo svojho rozsahu umožňuje. Hodnota bola vypočítaná na základe vzťahu (1.1). Vo vzťahu bolo počítané s hodnotou l = 100 mm, čo je prepokladaná vzdialenosť objektívu od povrchu objektov, ktoré sú predmetom inšpekcie. Ďalej vo vzťahu figurovala veľkosť plochy, ktorá je snímaná y = 60 mm, ktorá je dostatočne veľká, aby sa do nej zmestil najväčší objekt, na ktorom je vykonávaná inšpekcia (čierna súčiastka 49 mm).

Senzor umožňuje zachytávať snímky s rozlíšením 2592 x 1944 pixelov a snímať video v HD kvalite s rozlíšením 1920 x 1080 pixelov s frekvenciou 30 FPS. V nasledujúcej tabuľke 4.4 je uvedených ďalších päť formátov snímania videa a ich hodnoty FPS pre použitý senzor OV5647:

Formát	**Rozlíšenie**	**FPS**	**Metóda škálovania**
QSXGA (5 Mpx)	2592 x 1944	15	plné rozlíšenie
1080p	1920 x 1080	30	orezávanie
960p	1280 x 960	45	orezávanie, prevzorkovanie/zlučovanie
720p	1280 x 720	60	orezávanie, prevzorkovanie/zlučovanie
VGA	640 x 480	90	orezávanie, prevzorkovanie/zlučovanie
QVGA	320 x 240	120	orezávanie, prevzorkovanie/zlučovanie

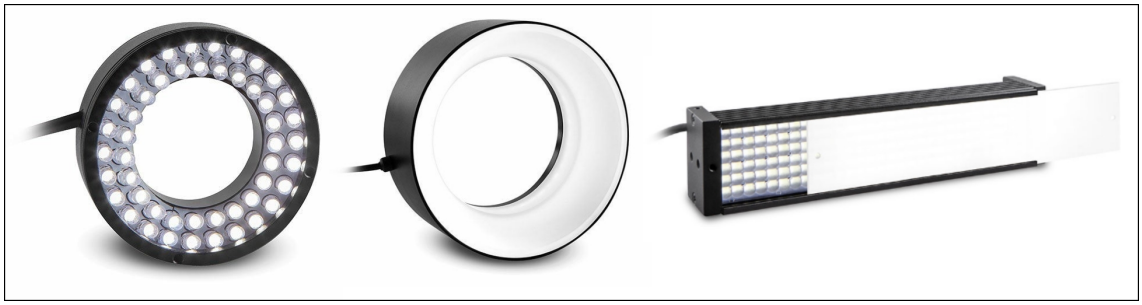
Tab. 4.4: Formáty a FPS snímateľných videí senzorom OV5647

4.3 Osvetľovač

V rámci výberu vhodných hardwarových prostriedkov pre vlastný systém strojového videnia, ktoré sú rozoberané v predošlých podkapitolách sa táto podkapitola zaoberá návrhom vhodného osvetlenia. Tak, ako bolo spomenuté v podkapitole 1.3.3, správne nasvietenie skúmaného objektu je pre celú aplikáciu rozpoznávania jednou z najdôležitejších úloh. V diplomovej práci bolo navrhnuté osvetlenie aplikované na trojicu objektov popísaných v kapitole 3. K dispozícii boli tri druhy osvetľovačov poskytnutých firmou *TE Connectivity*, konkrétne sa jedná o nasledujúce:

- **Kruhový osvetľovač s bielym LED svetlom**
- **Difúzny kruhový osvetľovač s bielym LED svetlom a nízkym uhlom vyžarovania**
- **Difúzny osvetľovač s plošným ovetľovacým poľom s bielym LED svetlom**

Všetky tri typy osvetľovačov, z ktorých bol vykonaný výber najvhodnejšieho osvetlenia, sú zobrazené na obrázku 4.3, pričom vľavo je kruhový osvetľovač, v strede je difúzny kruhový osvetľovač a vpravo je difúzny osvetľovač s plošným osvetľovacím poľom.



Obr. 4.3: Použité druhy osvetľovačov pri návrhu najvhodnejšieho osvetlenia [34] [35] [36]

Na každý z troch objektov je v rámci návrhu osvetlenia otestovaný vždy rovnaký typ osvetľovača alebo zmena pozície osvetľovača, čo je označené vždy príslušným písmenom, ako je vidieť na obrázkoch 4.4, 4.5, 4.6. Všetky objekty zobrazené na spomínaných obrázkoch sú fotené pomocou Raspberry Pi kamerovým modulom popísaným v podkapitole 4.2. V následujúcom odstavci je popísané priradenie označení písmen k príslušnému osvetľovaču spomínaných obrázkov.

Písmeno **a) značí kruhový osvetľovač**, ktorý bol počas osvetľovania umiestnený tak, aby jeho lúče dopadali kolmo na povrch objektu a objektív kamerového modulu

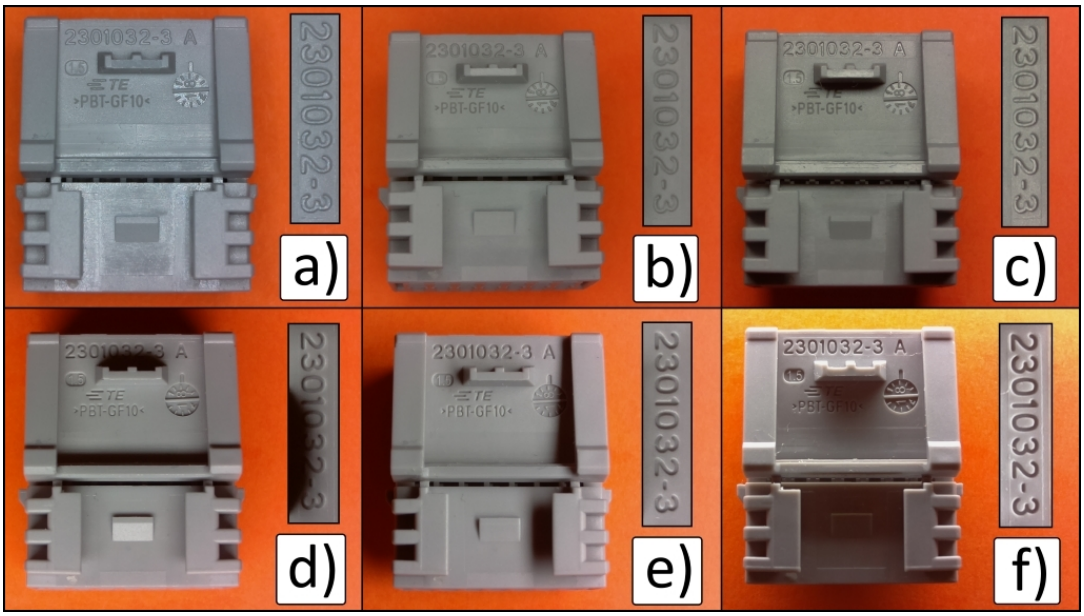
bol umiestnený v strede prstenca osvetľovača vo výške 10 cm nad povrchom objektu. Usporiadanie je ukázané v tabuľke 1.3 na spodnom obrázku. Písmeno **b) značí difúzny kruhový osvetľovač** umiestnený tak isto, ako v prípade a). Písmena **c)**, **d)**, **e)**, **f) značia difúzny osvetľovač s plošným osvetľovacím poľom** v rôznych jeho umiestneniach. V prípade c) osvetľovač ožaroval objekt akoby zvrchu obrázku, v prípade d) bol ožarovaný akoby zospodu obrázku, v prípade e) bol ožarovaný akoby sprava obrázku, všetky popísané prípady v priestore pod 45° uhlom tiež vo výške 10 cm a v prípade f) bol ožarovaný akoby zvrchu obrázku, ale s položeným osvetľovačom priamo na podklade, na ktorom je objekt položený vo vzdialenosti od seba 12 cm. Pri všetkých troch objektoch bol použitý podklad s oranžovou farbou, ktorá je univerzálna pre skúmaný šedý, hnedý a čierny objekt, pričom ich obrysy nesplynú s pozadím. V následujúcom odstavci je popísaný rozbor ožarovaných objektov z hľadiska kvality nasvietenia na obrázkoch porovnania, konkrétne záujmových oblastí s číslami šarží, ktoré sú predmetom rozpoznávania.

Na obrázku 4.4 šedého konektoru v prípade a) vnútorná plocha číslic jemne reflektuje dopadajúce svetlo, čím vzniká lesk a sú tým pádom zvýraznené aj hrany číslic s dostatočným množstvom pixelov. V prípade b) nie je reflektované žiadne svetlo z vnútornej plochy číslic, sú zložitejšie čitateľné a viac splývajú, ich obrysy obsahujú menšiu hrúbku, teda menej pixelov. Prípad c) je ukážkou zlého nasvietenia oblasti záujmu, cez číslice je vrhnutý tieň, čo podstatne zhoršuje následné možnosti rozpoznávania z obrazu.V prípade d) vzniká odraz resp. lesk vo vnútornej ploche číslic, keďže bolo osvetlenie akoby zvrchu obrázku, číslice majú rôzne hrúbky obrysov na hranách. V prípade e) nedopadá svetlo v dostatočnej miere na vnútornú plochu číslic do hĺbky výlisku, iba na plochu konektoru, tým pádom plocha ostáva tmavá, na základe čoho sú ale dobre zvýraznené hrany vylisovaných číslic. V prípade f) nedopadá žiadne svetlo do výlisku číslic, tiež iba na plochu konektoru, čo výborne zvýrazňuje rovnomerne celé číslice.

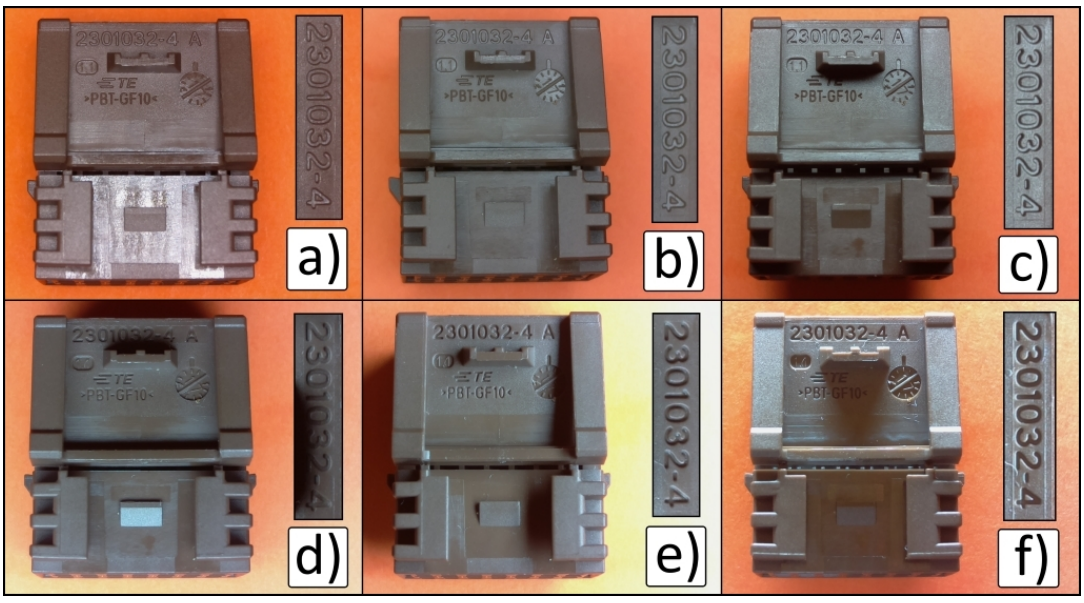
Na obrázku 4.5 sú vidieť veľmi podobné vlastnosti osvetlení vo všetkých prípadoch, ako tomu bolo pri rozbore osvetlení šedého konektoru až na malé drobnosti. V prípade b) pri použití difúzneho kruhového osvetľovača dochádza k odleskom od vnútornej plochy číslic, čo je spôsobené hnedou farbou konektoru. Pri šedom konektore odlesky splývajú s jeho farbou.

Na obrázku 4.6 v prípade a) súčiastka obsahuje vylisovanú drážku s lesklým povrchom a v nej vypuklé číslice. Drážka je pomerne veľká, a tak reflektuje veľa svetla, čo sa prejavuje jej bledým kontrastom na snímke. Číslice a hlavne ich hrany tak na obraze ostávajú tmavé (keďže sú vypuklé, môžeme ich brať ako nerovnosť povrchu)

lebo nereflektujú svetlo do takej miery. Takéto podmienky osvetlenia záujmovej oblasti s tmavými číslicami na bledom podklade sú žiaduce pre ďalšie spracovávanie a rozpoznávanie z obrazu. V prípade b) sú obrysy číslic rovnomerné a dostatočne hrubé, čiže stále rozpoznateľné z obrazu. V prípade c) sú týmto nasvietením dobre rozpoznateľné číslice z obrazu, tieň v drážke je minimálny. Prípad d) je opäť ukážkou zlého nasvietenia, cez číslice je vrhnutý tieň, čo je nežiaduce. V prípade e) sú ešte stále číslice rozpoznateľné z obrazu, pričom v prípade f) je snímok už moc presvet-



Obr. 4.4: Porovnanie šesť druhov osvetlení aplikovaných na šedý konektor



Obr. 4.5: Porovnanie šesť druhov osvetlení aplikovaných na hnedý konektor

lený, čo by si žiadalo výraznejšie predspracovanie obrazu.

Obr. 4.6: Porovnanie šesť druhov osvetlení aplikovaných na čiernu súčiastku

Na základe rozboru osvetlení podľa jednotlivých snímkov bolo vo vlastnej aplikácii strojového videnia navrhnuté použitie **kruhového osvetľovača s bielym LED svetlom**. Dôvodom je univerzálnosť osvetľovača pre použitie u všetkých troch typoch rozpoznávaných objektov, jeho jednoduchosť a možnosť jednoduchého upevnenia k mechanickej konštrukcií. U šedého a hnedého konektoru sa javil ako najpriaznivejší variant použitie difúzneho osvetľovača s plošným osvetľovacím poľom s bielym LED svetlom umiestneným podľa popisu f) priamo na základni. Naopak, u čiernej súčiastky tento variant vykazoval pomerne nízku rozlíšiteľnosť číslic. U šedého a hnedého konektoru bolo aj za použitia navrhnutého osvetľovača možné spoľahlivo a dostatočne vizuálne rozlišovať znaky.

5 SOFTWAROVÉ VYBAVENIE

5.1 Raspbian

Pre RPi existuje niekoľko operačných systémov, ktorými sú:

1. *Raspbian* - založený na distribúcií Debian, existujú tri verzie - prvá je Wheezy (Debian 7.0), pričom RPi 3 ju nepodporuje, druhá verzia je Jessie (Debian 8.0), ktorú nepodporuje najnovší model RPi 3+, tretia verzia je Stretch (Debian 9.0), ktorú podporujú všetky existujúce verzie RPi
2. *Pidora* - založené na distribúcií Fedora Remix
3. *RaspBMC* - založený na distribúcií Debian, má podporu od XBMC multimediálneho centra
4. *OpenELEC* - jedná sa o Linuxovú distribúciu s podporou XBMC multimediálneho centra
5. *Arch* - založený na distribúcií Arch Linux
6. *RISC OS* - nejedná sa o Linuxovú distribúciu, operačný systém navrhnutý iba pre ARM
7. *Windows 10 IoT Core* - založené na distribúcií Windows, pričom je táto verzia optimalizovaná pre embedded systémy, teda aj pre RPi modely 2 a 3
8. *Snappy Ubuntu Core* [37]

Pre účely práce bola zvolená ako najvhodnejšia distribúcia operačný systém ***Raspbian Stretch***, ktorý podporuje všetky verzie RPi, teda aj model 3. Tento systém bol vybraný hlavne z dôvodu, že sa jedná o primárny operačný systém pre RPi, čím by mala byť zaistená plná kompatibilita s hardwarom i s použitými aplikáciami. Keďže sa jedná sa o najpoužívanejší operačný systém, je možné s výhodou využiť skúsenosti ostatných užívateľov. Raspbian podporuje ako konzolový výpis, tak aj GUI. Oficiálne existujú dva druhy Raspbianu z hľadiska komplexnosti. Klasický Raspbian je plná distribúcia Debianu, ktorá obsahuje aj pracovnú plochu - GUI a predinštalované aplikácie na tvorbu textov, hry, editory na programovanie a iné. Raspbian Lite je osekaná distribúcia Debianu, ktorá obsahuje iba nevyhnutné balíky pre správnu funkciu jadra OS. Neobsahuje GUI a je ovládaný pomocou príkazového riadku.
Vo vlastnej práci je na RPi nainštalovaná úmyselne verzia *Raspbian Stretch Lite* z dôvodu šetrenia výkonu a priestoru. Všetky dostupné operačné systémy sú voľne dostupné z oficiálnych stránok Raspberry Pi vo formáte zip, ktorý po rozbalení obsahuje image operačného systému. Na SD karte môže byť vždy iba jeden operačný systém, po inštalácií nového sa automaticky starý premaže. V nasledujúcich kapito-

lách je popísaná inštalácia OS, prispôsobenie práce na RPi bez monitoru a úskalia, ktoré sa pri práci vyskytli.

5.1.1 Inštalácia

Ako už bolo spomenuté na konci predošlej kapitoly, operačný systém RPi je vždy uložený na (micro)SD karte, z ktorej bootuje. Minimálna doporučená veľkosť karty je 2 GB, čo môže byť ale nedostačujúce, keďže karta zároveň slúži aj ako pamäť dát. Vo vlastnom RPi je použitá až 32 GB microSD karta, ktorá svojou kapacitou plne dostačuje pre vlastnú aplikáciu. Úvodným úkonom je vytvorenie bootovateľnej microSD karty s OS Raspbian. Za týmto účelom bol za pomoci OS Windows použitý nástroj *Rufus*, v ktorom stačí vybrať cieľové zariadenie, na ktoré bude nahraný cieľový OS a zvoliť **.img* súbor s týmto OS. Potom stačí dať tlačidlo ŠTART a OS sa nahrá. Bezprostredne po nahraní Raspbianu je karta rozdelená na dve partície FAT a Ext3. K rovnakému účelu je možné použiť aj nástroj *Win32 Disk Imager* (tiež pod OS Windows), ktorý ponúka aj vytváranie záloh celej pamäťovej karty pomocou tlačidla Read. To je výhodné predovšetkým v prípadoch veľkých zásahov do systému, ako napríklad v jadre. Výhodou takto získaného image karty je, že nie je potrebné nič znovu inštalovať, stačí kartu s nahraným image vložiť do pôvodného RPi alebo úplne iného a všetko funguje tak, ako v bode, kedy bola vytvorená záloha. Táto vlastnosť je v práci často využívaná, hlavne pri vykonávaní zásadnych zmien alebo po dosiahnutí významných úspechov pri práci na vlastnej aplikácii.

5.1.2 Základné nastavenia, prvotné spustenie

Pri prvom spustení sa doporučuje RPi pripojiť cez HDMI konektor k monitoru, pripojiť klávesnicu, prípadne myš. Po nabootovaní OS sa spustí príkazový riadok, kde je potrebné vykonať prihlásenie cez užívateľské meno: pi a heslo: raspberry. Po prihlásení je potrebné vykonať základné nastavenie príkazom:

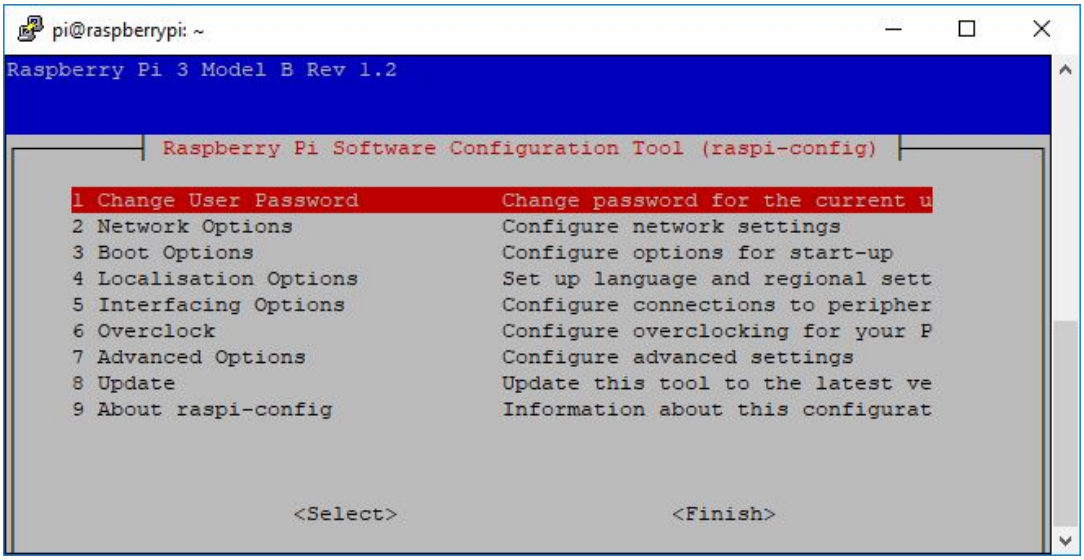
```
> sudo raspi-config
```

Tým sa otvorí ponuka s hlavnými nastaveniami RPi (pozri obrázok 5.1), kde sa nastavuje prvým príkazom (doporučené nastavenie) zmena hesla, aby sa do RPi nedostal prípadne cudzí užívateľ na sieti. Druhým príkazom sa nastavuje sieťová konfigurácia, konkrétne hostname a automatické pripájanie RPi na Wi-Fi sieť po zadaní SSID a hesla siete. Tretím príkazom sa nastavuje bootovanie pri spustení. Štvrtým nastavenie jazyka, kde je implicitná angličtina, ďalej rozloženie klávesnice s časovým pásmom. V piatom príkaze sa po jeho zakliknutí dostaneme do rozširujúceho menu, kde sa povoľujú alebo zakazujú jednotlivé hardwarové periférie a

protokoly RPi. Jednou z nich je aj kamera, o nastavení ktorej bude písané v kapitole 5.2 a povolenie SSH protokolu (popísané v kapitole 5.1.3). Ďalším príkazom sa nastavuje pretaktovanie procesoru, ktoré beží štandardne na 1,2 GHz pomocou *config.txt* konfiguračného súboru (nachádza sa v adresári *boot* a je k nemu potrebné pristupovať s právami roota). Pre vyššie frekvencie sa však doporučuje použitie aspoň pasívnych chladičov na RPi. V siedmom príkaze v rozširujúcom menu je príkaz `Expand Filesystem`, ktorý bol spustený ako prvý hneď na začiatku, lebo Raspbian automaticky nenastavuje MicroSD kartu na plnú veľkosť. Ostatné príkazy pre vlastné účely aplikácie nie sú podstatné. Po pripojení RPi k internetu a reštartovaní bol systém aktualizovaný z príkazového riadku následujúcimi príkazmi:

```
> sudo apt-get update
> sudo apt-get upgrade
```

Dvojicu týchto príkazov je potrebné vždy použiť pred inštaláciou programov. RPi má implicitne nastavenú DHCP IP adresu ako pre eth0 tak pre wlan.



Obr. 5.1: Konfiguračné menu Raspberry Pi

5.1.3 Vzdialená správa, prenos súborov

RPi povoľuje vzdialené pripojenie cez SSH protokol, ktorý slúži pre zabezbečený prístup cez príkazový riadok z iného počítača. Pre pripojenie cez tento protokol je možné použiť program *PuTTy*, kde sa zadáva iba IP adresa pridelená RPi. Po zadaní prihlasovacích údajov sa sprístupní príkazový riadok pre RPi. SSH protokol je implicitne v nastaveniach zapnutý. V menu `raspi-config` ho ide vypnúť v príkaze

`Advanced Options`.

Pre účely testovania osvetlenia s kontinuálnym zobrazením snímaného obrazu RPiCam na monitore a zobrazovania priebežných výsledkov pri vývoji aplikácie rozpoznávania bolo potrebné doinštalovať GUI pre RPi príkazom:

```
> sudo apt install raspberrypi-ui-mods
```

Pre možnosť zobrazenia GUI pomocou VNC bolo potrebné v menu `sudo raspi-config` v možnosti `Interfacing Options` nastaviť `VNC` na `Enable`. Pre pripojenie sa ku GUI RPi bol využitý nástroj *VNC Viewer*, nainštalovaný na OS Windows. Vďaka doinštalovaniu GUI osobitne, bežiacim pod Raspbian Lite nebolo potrebné inštalovať Raspbian s GUI, ktorý obsahuje veľa nepotrebných predinštalovaných aplikácií, ktoré by zaberali pamäť.

Počas práce s RPi a pri vývoji vlastnej aplikácie bol často využívaný program *WinSCP* na prenos súborov medzi RPi a osobným počítačom. Využíva k tomu SFTP a SCP protokol, disponuje užívateľsky jednoduchým a intuitívnym grafickým prostredím.

5.2 Raspberry Pi kamerový modul

Pre použite RPiCam modulu je nutná ako prvá jeho aktivácia resp. povolenie v základnom nastavení `raspi-config`. Tým je kamerový modul sprístupnený pre prácu z príkazového riadku. Pre obstaranie testovacieho snímku a videa kamerou pomocou príkazového riadku sa použijú príkazy:

```
> raspistill -o testovaci_snimok.jpg
> raspivid -o testovacie_video.h264 -t 5000
```

Snímok a video sú uložené do aktuálneho adresára (za predpokladu, že má užívateľ právo zapisovať do adresára) pod názvom testovaci_snimok.jpg a testovacie_video.h264 o dĺžke 5000 ms.

Vo vlastnej aplikácii je potrebné pristupovať však ku kamere cez jazyk Python, konkrétne vo verzií Python 3, v ktorej je aplikácia rozpoznávania naprogramovaná. Pre sprístupnenie kamery v tejto verzii je potrebná inštalácia nasledujúceho balíku, za ktorým nasleduje otestovanie funkčnosti kamery jednoduchým príkazom:

```
> sudo apt-get install python-picamera python3-picamera
> python3 -c "import picamera"
```

Ak pri otestovaní konzola nezahlási chybu, kamera je prístupná z Python 3.

5.2.1 Dosahovaná veľkosť FPS Raspberry Pi kamery

Skutočné dosahované veľkosti FPS kamery implementáciou popísanou v kapitole 7.1, bez prítomnosti samotného programu rozpoznávania, boli zmerané veľkostí FPS kamery ukázané v tabuľke 5.1.

Použité rozlíšenie kamery	**Veľkosť FPS kamery**
2592x1944	3
1920x1440	**5.2**
1920x1080	5.5
1280x960	5.5
1280x720	43
640x480	86
320x240	86

Tab. 5.1: Reálne veľkosti zmeraných FPS Raspberry Pi kamery

Vo vlastnej aplikácii rozpoznávania je pracované s rozlíšením 1920x1440, vďaka ktorému je snímok kvalitný a zvyšuje sa tak úspešnosť rozpoznávania.

5.3 Knižnica OpenCV

OpenCV patrí medzi najrozšírenejšie knižnice pre spracovanie obrazu. Vývoj knižnice OpenCV započala v roku 1999 spoločnosť *Intel*. Prvá verzia 1.0 bola vydaná v roku 2006. Druhá verzia 2.0 vydaná v roku 2009 bola prevratná, obsahovala množstvo nových modulov a funkcií optimalizovaných z hľadiska výkonu. Od roku 2016 prebrala vývoj OpenCV spoločnosť *Itseez*. Oficiálne vydania knižnice sa objavujú každých šesť mesiacov. Najnovšia verzia OpenCV je 4.1.0 z roku 2019.
Knižnica, hlavne vďaka veľkému množstvu základných funkcií, ktorých použitie patrí medzi pomerne jednoduché pre užívateľa. Knižnica je "open source" a obsahuje veľa funkcií a algoritmov pre transformáciu obrazu, úpravu formátu, analýzu obrazu a filtrovanie, detekciu objektov v obraze, prácu s videom a užívateľskými oknami a funkcie pre kalibráciu kamery. Niektoré funkcionality a syntax sa podobajú zápisu v Matlabe, dokonca prístup k maticiam je obdobný ako v Matlabe. Výhodou knižnice je jednoduchá komunikácia so štandardným hardware, ktorým je aj RPi kamera,

ktorú nie je potrebné zložito nastavovať. Knižnica využíva k tomuto účelu systémové ovládače. Knižnica má podporu programovacích jazykov C, C++, Java a Python a môže bežať pod OS Linux, Windows, Mac OS X a Android. Vo vlastnej aplikácii rozpoznávania je využívaná verziu knižnice OpenCV 3.4.4. [38]
Pred samotnou inštaláciou knižnice je potrebné doinštalovať ešte potrebné súčasti:

```
> sudo apt-get install build-essential cmake unzip pkg-config -y
> sudo apt-get install libjpeg-dev libpng-dev libtiff-dev -y
> sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev
libv4l-dev -y
> sudo apt-get install libxvidcore-dev libx264-dev -y
> sudo apt-get install libgtk-3-dev -y
> sudo apt-get install libcanberra-gtk* -y
> sudo apt-get install libatlas-base-dev gfortran -y
> sudo apt-get install python3-dev
> sudo apt install libavcodec-extra57
> sudo apt install libjasper1
> cd
```

Následne je potrebné stiahnúť samotnú knižnicu OpenCV 3.4.4 a contrib moduly z GitHubu:

```
> wget -O opencv.zip https://github.com/opencv/opencv/archive/3.4.4.zip
> get -O opencv_contrib.zip https://github.com/opencv/opencv_contrib/
archive/3.4.4.zip
> unzip opencv.zip
> mv opencv-3.4.4 opencv
> mv opencv_contrib-3.4.4 opencv_contrib
```

Ďalej je potrebné nainštalovať Numpy:

```
> sudo apt-get install python3 python3-setuptools python3-dev -y
> wget https://bootstrap.pypa.io/get-pip.py
> sudo python3 get-pip.py
> sudo pip3 install numpy
```

Potom treba zadať atribúty kompilácie a inštalácie OpenCV 3.4.4:

```
> cd  /opencv
```

```
> mkdir build
> cd build
> cmake -D CMAKE_BUILD_TYPE=RELEASE
-D CMAKE_INSTALL_PREFIX=/usr/local
-D OPENCV_EXTRA_MODULES_PATH= /opencv_contrib/modules
-D ENABLE_NEON=ON
-D ENABLE_VFPV3=ON
-D BUILD_TESTS=OFF
-D OPENCV_ENABLE_NONFREE=ON
-D INSTALL_PYTHON_EXAMPLES=OFF
-D BUILD_EXAMPLES=OFF ..
```

Kompiláciu knižnice OpenCV je možné spustiť aj na všetkých štyroch jadrách RPi nastavením premennej `CONF_SWAPSIZE=2048` v súbore `/etc/dphys-swapfile`, čím sa oveľa rýchlejšie knižnica skompiluje. Po nastavení premennej na danú hodnotu, bolo ešte potrebné aktualizovať hodnotu premennej pomocou príkazov:

```
> sudo /etc/init.d/dphys-swapfile stop
> sudo /etc/init.d/dphys-swapfile start
```

Nakoniec sa spustí kompilácia a následná inštalácia príkazmi:

```
> make -j4
> sudo make install
> sudo ldconfig
```

Po úspešnej kompilácií a inštalácií knižnice bola ešte hodnota premennej nastavená späť na `CONF_SWAPSIZE=100` a opäť aktualizovaná príkazmi. Nakoniec ešte boli doinštalované riadiace moduly pre Python 3:

```
> sudo pip3 install opencv-python
> sudo pip3 install opencv-contrib-python
```

Celá kompilácia knižnice aj s inštaláciou trvala zhruba jednu hodinu.

5.4 Tesseract OCR

Tesseract OCR engine je software určený pre optické rozpoznávanie znakov pre rôzne operačné systémy. Bol pôvodne vyvíjaný v *Hewlett-Packard Laboratories* od roku

1985, v roku 2005 bol prepísaný do C++ a sprístupnený ako "open source", od roku 2006 pokračuje vo vývoji spoločnosť *Google*. Posledná stabilná verzia 3.05 vyšla v roku 2017, verzie 4.0 a vyššie sú zatiaľ beta verzie.
Tesseract dokáže rozpoznávať cez 100 jazykových sád v základnej konfigurácii. Keďže sa jedná o engine, používa sa cez príkazový riadok. Od verzie 4.0 sa využíva Tesseract engine na báze hlbokého učenia pomocou konvolučných neurónových sietí, vďaka čomu dosahuje výrazne lepšiu presnosť rozpoznania než predošlé verzie, výmenou ale za vyššiu výpočtovú náročnosť. Súčasné trénované dáta pre neurónovú sieť Tesseractu sa pohybujú okolo 400000 riadkov textu v 4500 druhoch fontov. Aj keď je množstvo trénovaných dát dostačujúce, užívateľ má možnosť pre svoju konkrétnu úlohu neurónovú sieť trénovať na vlastných dátach, čím zvýši presnosť rozpoznania. [39]
Pred aplikovaním Tesseractu na cieľový obraz s textom musí byť tento obraz najskôr predspracovaný. Po predspracovaní by mal obraz spĺňať pre efektívne rozpoznanie textu nasledujúce body:

1. výška znakov v obraze musí byť aspoň 20 px
2. text nesmie byť natočený
3. text nesmie byť skosený
4. oblasť pre rozpoznanie textu by mala obsahovať iba text s jeho malým okolím a s minimálnym množstvom šumu

Vo vlastnej aplikácii rozpoznávania je použitý Tesseract 4.0.0, jeho inštalácia na Rpi prebehla pomocou nasledujúcich príkazov:

```
> git clone https://github.com/thortex/rpi3-tesseract
> cd rpi3-tesseract/release
> ./install_requires_related2leptonica.sh
> ./install_requires_related2tesseract.sh
> ./install_tesseract.sh
```

5.5 Databáza MariaDB

MariaDB je relačný databázový systém, ktorý je vyvíjaný ako nástupnícka vetva pôvodného databázového systému MySQL. MariaDB je vyvíjané od roku 2009 a hlavným dôvodom k vytvoreniu tejto vetvy bola obava pôvodných vývojárov MySQL o ďalší osud a smerovanie tohto softwaru po jeho odkúpení spoločnosťou *Oracle*, či bude naďalej udržaná licencia slobodného softwaru.[40]

Vo vlastnej aplikácii rozpoznávania sa používa databáza MariaDB pre zápis výsledkov rozpoznaného výrobného čísla konektoru spolu s ďalšími informáciami. Aby však bolo možné pracovať s databázou bežiacou na RPi, bolo potrebné nainštalovať na OS Raspbian MariaDB Server nasledujúcim príkazom do konzoly:

```
> sudo apt-get install mariadb-server
```

Následne bola pomocou príkazov v konzole vytvorená databáza s názvom `TE_Connectors`, bol vytvorený nový užívateľ a boli mu nastavené práva:

```
> sudo mysql -u root -p -h localhost
> CREATE DATABASE TE_Connectors;
> USE TE_Connectors;
> CREATE USER 'pi'@'localhost' IDENTIFIED BY 'martin';
> FLUSH PRIVILEGES;
> sudo service mysql restart
```

Ďalej bol nainštalovaný MySQL klient, aby fungovalo importovanie knižnice `MySQLdb` do Pythonu:

```
> sudo pip3 install mysqlclient
```

Nakoniec bolo ešte potrebné nastaviť, aby sa po pripojení RPi do siete a priradení DHCP IP adresy išlo na túto adresu vždy pripojiť bez manuálneho prepisovania aktuálnej pridelenej adresy v konfiguračnom súbore databázy. Toho bolo docielené dopísaním nasledujúcich dvoch riadkov do konfiguračného súboru nachádzajúceho sa v `/etc/mysql/my.cnf`:

```
[mysqld]
bind-address = 0.0.0.0
```

Tým pádom MariaDB Server vysiela na všetkých IP adresách a po pripojení na DHCP IP adresu je možné sa na databázu pripojiť. Po pripojení bola vytvorená nová tabuľka `Connectors` pomocou nástroja *HeidiSQL* so štyrmi stĺpcami. Do prvého stĺpca s názvom `Time_stamp` je ukladaná časová známka zapísania výsledku rozpoznania, do druhého stĺpca s názvom `Connector_type` je ukladaný typ rozpoznaného konektoru, do tretieho stĺpca s názvom `Connector_number` je ukladané rozpoznané výrobné číslo konektoru a do štvrtého stĺpca s názvom `Processing_time` je ukladaný celkový procesný čas od zdetegovania konektoru až po zápis do databázy.

Na obrázku 5.2 je ukážka zapísaných výsledkov do databázy zobrazených pomocou nástroja *HeidiSQL*.

Hostiteľ: 192.168.0.25 | Databáza: TE_Connectors | Tabulka: Connectors | Dáta | Dotaz

TE_Connectors.Connectors: 24 řádků celkem (přibližně)

Time_stamp	Connector_type	Connector_number	Processing_time
2019-04-24 00:52:45	Brown	2301032-4	2,6
2019-04-24 00:52:57	Brown	2301032-4	2,65
2019-04-24 00:53:33	Brown	2301032-4	2,64
2019-04-24 00:53:41	Brown	2391032-4	2,66
2019-04-24 00:54:04	Brown	2301032-4	2,62
2019-04-24 00:54:21	Brown	2301032-4	3,01
2019-04-24 00:54:29	Brown	2301032-4	2,9
2019-04-24 00:54:39	Brown	2301032-4	2,65

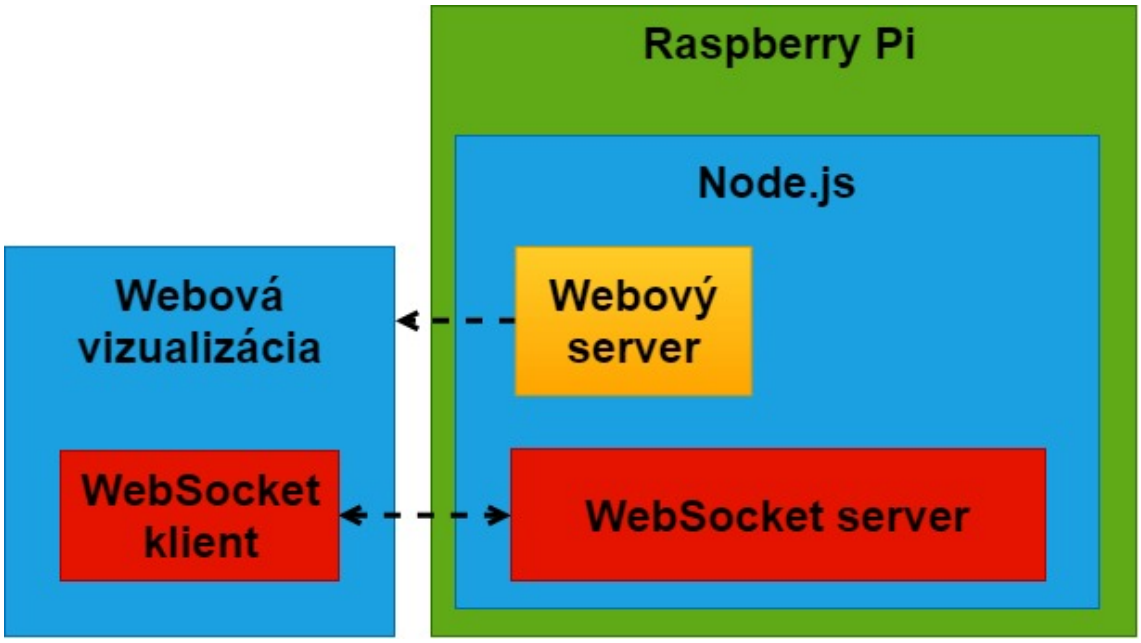
Obr. 5.2: Ukážka zapísaných výsledkov do databázy

5.6 WebSocket

WebSocket je webová technológia umožňujúca obojstrannú komunikáciu *(angl. full duplex)* medzi klientom a serverom cez jedno TCP spojenie, pričom nahradzuje protokol HTTP, ktorý neumožňuje obojstrannú komunikáciu. Komunikačný protokol WebSocket bol štandardizovaný v roku 2011 komisiou *IETF*, ktorá vyvíja a podporuje internetové štandardy. WebSocket je navrhnutý k implementácii vo webových prehliadačoch a webových serveroch, ale môže byť využitý akoukoľvek klientskou či serverovou aplikáciou. WebSocket protokol je nezávislý na TCP protokole.
Výhodou WebSocket protokolu je komunikácia v reálnom čase a malé zaťaženie siete. V súčasnosti je podporovaný najpoužívanejšími webovými prehliadačmi - Google Chrome, Mozzila Firefox, Internet Explorer, Opera, Safari. [42]

5.6.1 Knižnica Socket.io

Socket.io je knižnica určená pre komunikáciu webových aplikácií v reálnom čase, primárne používa pre komunikáciu WebSocket protokol a je napísaná v jazyku JavaScript. Knižnica je rozdelená na dve časti, klientská časť beží vo webovom prehliadači, serverová časť na serveri Node.js. Knižnica Socket.io má architektúru riadenú udalosťami *(angl. events)*. Jednou z mnohých výhod knižnice je podpora viacerých inštancií servera. [43]
Popísaná knižnica je použitá vo vlastnej aplikácii rozpoznávania, sprostredkováva tzv. frontend pre užívateľa vo forme webovej stránky, na ktorej sú vizualizované aktuálne výsledky a náhľad rozpoznávania výrobných čísiel konektorov. Bližšie je popísaná webová vizualizácia v kapitole 8.
Obrázok blokovej schémy popisujúci použitie knižnice Socket.io vo vzťahu klient-server vo vlastnej aplikácii je ukázaný na obrázku 5.3.



Obr. 5.3: Koncepcia knižnice Socket.io použitej vo vlastnej aplikácii vo vzťahu klient-server

Vizualizácia formou webovej stránky bola zvolená zámerne, aby spĺňala novodobé štandardy priemyselného prostredia, v ktorom tvorba vizualizácie u automatizačných firiem postupne smeruje k webovým stránkam. Veľkou výhodou takejto vizualizácie je možnosť jednoduchého pripojenia k nej pomocou podporovaného webového prehliadača so zariadením s ľubovoľným operačným systémom. Typicky pomocou chytrých telefónov, tabletov alebo podobných zariadení. Webová vizualizácia so svojimi výhodami prekonáva vizualizácie, ku ktorým sa klient pripája pomocou grafického programu VNC, ktorý musí byť na klientskom zariadení nainštalovaný a serverové zariadenie s vizualizáciou musí vytvárať VNC server. Naviac, pri vykresľovaní vizualizácie je viac výkonovo zaťažené serverové zariadenie.

Aby bolo možné s knižnicou Socket.io pracovať v RPi, boli zadané nasledujúce inštalačné príkazy do konzoly:

```
> sudo pip install python-socketio
> sudo pip install aiohttp
```

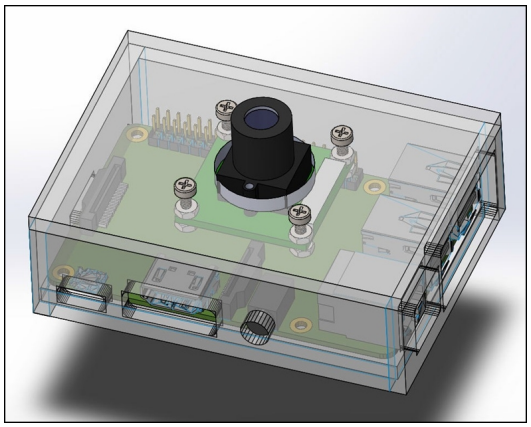
Bližšie popísaná vlastná implementácia algoritmu pomocou knižnice Socket.io je popísaná v kapitole 7.3.2.

6 MECHANIKA PRÍPRAVKU

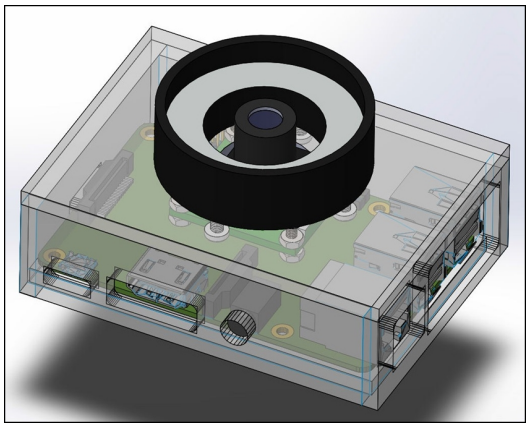
Pre vytvorenie úspešnej aplikácie strojového videnia je nutné správne uchytenie kamerového modulu do požadovanej výšky nad objektom, na ktorom je vykonávaná inšpekcia kamerou v súčinnosti s osvetlením.
Pre tento účel bola navrhnutá jednoduchá mechanická konštrukcia, ktorá všetky tieto požiadavky spĺňa. Aplikácia strojového videnia vykonáva inšpekciu OCR konektorov a súčiastky experimentálne, nie je zaradená do skutočného výrobného procesu, čiže na návrh mechanickej konštrukcie neboli kladené žiadne konkrétne požiadavky. V konečnom dôsledku to znamená, že navrhnutá konštrukcia je čo najjednoduchšia a zároveň praktická, pri jej zhotovení nie je potrebný žiadny výrobný proces, je poskladaná z bežných profilov.
Konštrukcia je navrhnutá v tvare portálu z hliníkových miniTec profilov o priereze 30x30 mm a o dĺžke 35 cm. Jednotlivé profily sú pospájané pomocou uhlových spojovacích dielov. Základová báza má oranžovú farbu, ktorá výborne kontrastuje s rozpoznávanou súčiastkou, ktorá je na nej umiestnená. Na nosnom profile je upevnená krabička obsahujúca RPi a RPiCam v jednom. Celá krabička je uchytená na spodnej hrane nosníku portálu tak, aby vyčnievajúci objektív kamery von z krabičky smeroval nadol, čiže snímal základňu. Okolo objektívu je na krabičke upevnený aj prstencový osvetľovač. Bližší popis prevedenia je rozoberaný ďalej v podkapitole 6.1. Výška nosníku je nastaviteľná podľa potreby povolením dvoch protiľahlých skrutiek.

6.1 Púzdro pre Raspberry Pi a Raspberry Pi kameru

Ako už bolo spomenuté v predošlej kapitole, RPi je spolu s RPiCam modulom zapuzdrené v spoločnej krabičke. Pre ukážku poskladania zariadení do krabičky bol vytvorený model so skutočnými rozmermi. Model bol vytvorený v konštrukčnom programe *SolidWorks 2018*. Transparentná krabička pre RPi je zakúpená, pričom je na jej vrchnej strane v strede vlastnoručne navŕtaná diera s priemerom 23 mm a okolo nej štyri montážne diery s priemerom 2 mm. Cez stredovú dieru sa prevlečie celá objímka spolu s objektívom kamery, aby vyčnievali. Pomocou montážnych dier so skrutkami a maticami M2 sa uchytí RPiCam ku krabičke. Prevedenie je ukázané na obrázku 6.1. Upevnenie kruhového osvetľovača ku krabičke v osi objektívu v strede je realizované pomocou štyroch skrutiek M3. Model prevedenia je ukázaný na obrázku 6.2.



Obr. 6.1: Model upevnenia Raspberry Pi kamery

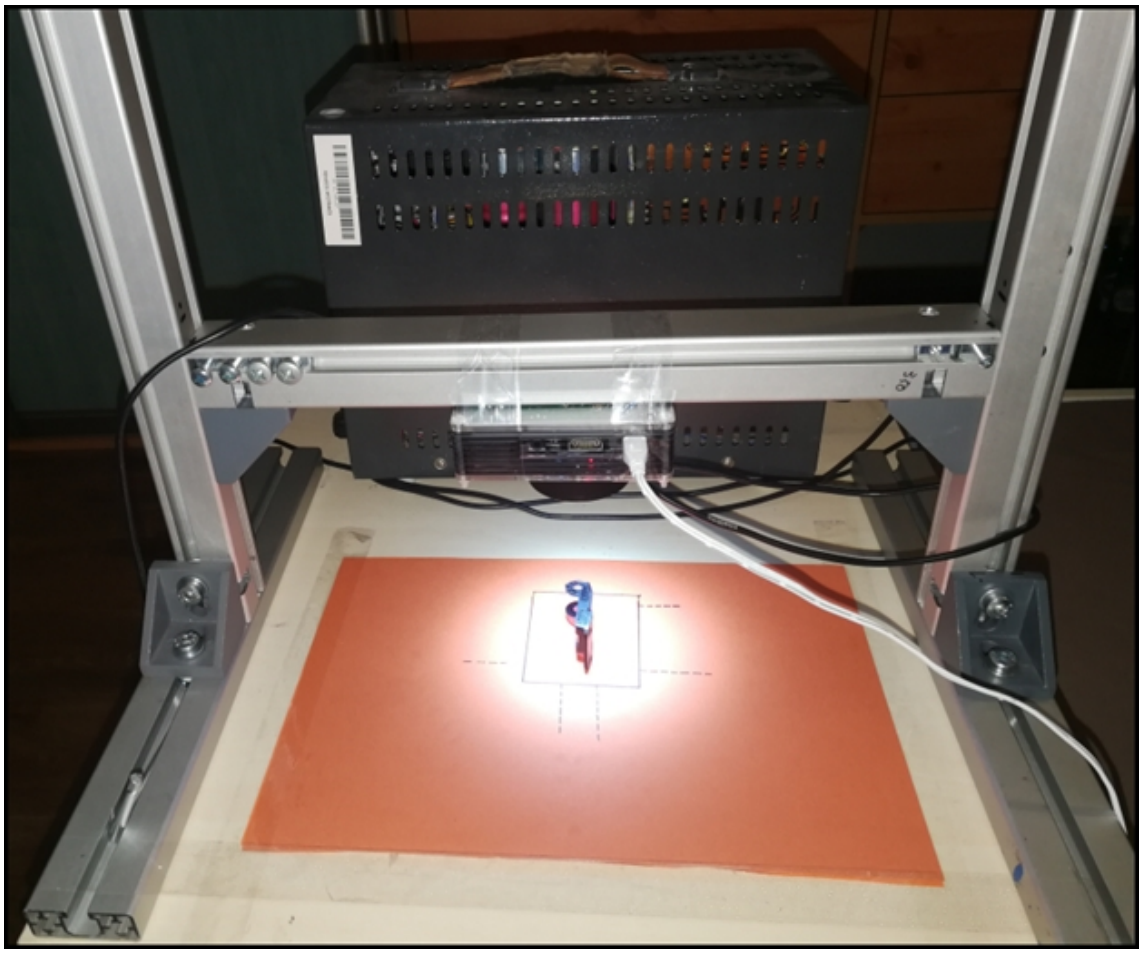


Obr. 6.2: Model upevnenia Raspberry Pi kamery spolu s osvetľovačom

Na obrázku 6.3 je ukážka reálne vytvoreného kompaktného zariadenia pre inšpekciu čísla konektorov podľa popísaného modelu.

Obr. 6.3: Zariadenie pre inšpekciu čísla konektorov

Na obrázku 6.4 je ukážka mechanickej konštrukcie s upevneným zariadením pri vykonávaní inšpekcie čísla konektoru.



Obr. 6.4: Mechanická konštrukcia s upevneným zariadením pri vykonávaní inšpekcie

7 ALGORITMUS INŠPEKCIE OCR

V kapitolách nižšie je detailne popísaný algoritmus vlastného inšpekčného programu. Ten je implementovaný v cyklicky bežiacom Python skripte `recgApp.py`. Pre inicializáciu rozlíšenia kamery, veľkosti FPS kamery, počet používaných vlákien procesoru, veľkosti konektorov v pixeloch, tolerancie a pre ďalšie preddefinované hodnoty ohľadne rozpoznávaných konektorov bol vytvorený súbor s konštantami `constants.py`, z ktorého sú čerpané hotnoty v hlavnom programe. Pri popise vlastných funkcií v kapitolách nižšie je vždy zobrazený prototyp funkcie so vstupnými parametrami namiesto funkčného volania, lebo niektoré funkcie sú volané z viacerých miest programu s rôznymi vstupnými parametrami. Funkcie, ktoré majú predponu cv2, sú použité z knižnice OpenCV a je zobrazené ich funkčné volanie s obecnými názvami parametrov podľa návodov k tejto knižnici.

7.1 Získavanie snímkov z kamery

Existuje viacero spôsobov, ako získavať, ukladať a predspracovať snímky z RPi kamery. Prvým a zároveň najjednoduchším z nich je ukladať vždy aktuálny snímok do určeného adresára, udržiavať si frontu napr. piatich odfotografovaných snímkov za sebou, na ktorých sa vykonáva predspracovanie. Nakoniec v adresári posledný vždy vymazávať. Tento spôsob je ale neefektívny, lebo operácia zapisovania súboru do adresára je časovo náročná, čiže pre vlastnú aplikáciu inšpekcie je nepoužiteľný. Druhým, sofistikovanejším spôsobom, je ukladanie snímkov do streamu v pamäti RPi, čím sa nezaberá miesto na microSD karte a šetrí sa čas, lebo sa zapisuje priamo do pamäti a nie do adresára. Popísané dva spôsoby využívajú pri svojej činnosti iba jedno jadro procesoru. Použitím viacvláknovej architektúry programu *(angl. multithreading)*, ktorá je vlastnosťou procesoru, je možné využiť všetky štyri jadrá procesoru RPi. To znamená, že je možné využiť zvyšných 75 % nevyužitého výkonu RPi. Tretím a zároveň najsofistikovanejším spôsobom získavania a predspracovania snímkov z kamery je rozdeliť operáciu získavania snímkov a predspracovania do samostatných vlákien. Ideálne, operácii získavania snímkov z RPi kamery prideliť jedno vlákno, operácii predspracovania snímkov, ktorá je výpočtovejšie náročnejšia, zvyšné vlákna. Týmto spôsobom je možné dosahovať vysoké FPS RPi kamery a využiť plný potenciál výpočtového výkonu RPi. Tento spôsob je použitý vo vlastnom inšpekčnom programe a je bližšie popísaný v nasledujúcich dvoch podkapitolách.
Veľkosť rozlíšenia snímkov kamery vo vlastnej aplikácii rozpoznávania je nastavený v súbore s konštantami na pevnú hodnotu 1920x1440. Pomerne vysoké rozlíšenie snímkov je zvolené zámerne pre zvýšenie efektivity rozpoznávania.

7.1.1 Vlákno pre získavanie a ukladanie snímkov do bufferu

Pre získavanie snímkov z RPi kamery sa používa trieda `ImageCapture`, ktorá pracuje v samostatnom vlákne. V tejto triede sa nachádza metóda `run()`, ktorá vždy vezme najstaršie nepoužívané vlákno zo zoznamu čakajúcich vlákien (processorPool) a zachytí nasledujúcu snímku kamery pomocou funkcie `cap.read()`, ktorú hneď odošle do procesoru aj s upozornením o udalosti. Ak sa v zozname čakajúcich vlákien nenachádza žiadne nepoužívané vlákno, počká sa 10 ms a kontroluje sa znovu.

7.1.2 Vlákna pre predspracovanie snímkov

Pre predspracovanie získaných snímkov sa používa trieda `ImageProcessor`, ktorá dostáva vždy upozornenie na udalosť o získaní nového snímku. Po upozornení sa volá metóda `run()`,v ktorej si vždy snímok preberá jedno z dvoch voľných vlákien (počet vlákien určených na predspracovanie môže byť aj menej do minima jedného voľného vlákna, pomocou nastavenia konštanty `PROG_THREADS_NUM`) celkového množstva použitých vlákien vo vlastnej aplikácii. Vlákno zapíše snímok do streamu a následne je navrátené naspäť do zoznamu čakajúcich vlákien.V metóde `ProcessImage()`, ktorá je určená priamo pre predspracovanie obrazu, je snímka vytiahnutá zo streamu a ďalej spracovávaná funkciami popísanými v nasledujúcich kapitolách. Zdrojový kód pre získavanie snímkov z kamery spolu s predspracovaním snímkov pomocou vlákien je použitý z [41] a modifikovaný pre vlastné potreby aplikácie rozpoznávania.

7.2 Detekcia konektoru v obraze a rozpoznanie výrobného čísla

Všetky následujúce podkapitoly popisujúce funkcie algoritmu sú zoradené za sebou postupne, ako sú volané v programe inšpekcie `recgApp.py`. V poslednej podkapitole 7.1 sú názorne ukázané jednotlivé operácie popísaných funkcii v obrázkovom diagrame.

7.2.1 Predspracovanie snímkov - ImagePreprocess

Získaný aktuálny snímok z kamery, ktorý bol predaný vláknu určenému pre predspracovanie obrazu, je ďalej predspracovaný do vhodnej formy pomocou viacerých morfologických a výpočtových operácií. Pre tento účel bola vytvorená funkcia `ImagePreprocess(image)`, ktorej vstupom je aktuálny snímok z kamery prevedený do odtieňov šedej a výstupom je hranovaný obraz, ktorý obsahuje vyplnený objekt v obraze bielou farbou pixelov. V tele funkcie sú vykonané následujúce operácie za

sebou:

1. na šedotónový obrázok je aplikovaný Cannyho hranový detektor pomocou funkcie `cv2.Canny(image, edges, threshold1, threshold2, aperture_size = 3)`, z ktorej je výstupom hranovaný obrázok
2. na hranovaný obrázok sa aplikuje negácia pomocou funkcie `cv2.bitwi se_not(image)`, čím je na výstupe získaný obrázok s bielou farbou kontúr a čiernym pozadím
3. na negovaný hranovaný obrázok sa aplikuje morfologické uzavretie pomocou funkcie `cv2.morphologyEx(image, cv2.MORPH_CLOSE, kernel)`, ktorého výstupom je obrázok so zaplnenými medzerami bielymi pixelmi v objekte
4. na morfologicky uzavretý obrázok sa aplikuje morfologická dilatácia pomocou funkcie `cv2.dilate(image, kernel, iterations = 3)`, ktorá sa vykoná tri iterácie a objekt ešte viac vyplní bielymi pixelmi
5. na morfologicky dilatovaný obrázok je aplikovaná operácia záplavového plnenia pomocou funkcie `cv2.floodFill(image, mask, seedPoint, newVal, loDiff, upDiff, flags)`, ktorá vyplní bielymi pixelmi celý objekt až po jeho hrany, ktoré su hranicou medzi objektom a pozadím
6. zo zaplaveného obrázku sa vykoná negácia (vznikne obrázok s doposiaľ nevyplnenými miestami v objekte) a následne takto negovaný zaplavený obrázok je bitovo sčítaný s pôvodným zaplaveným obrázkom z čoho vznikne plne vyplnený objekt.

Všetky popísané operácie sú cyklicky vykonávané na všetkých získaných snímkoch z kamery.

7.2.2 Kontrola prítomnosti konektoru - CheckArea

Pre kontrolu prítomnosti objektu, resp. konektoru v obraze, bola vytvorená funkcia `CheckArea(imageReady, imageGrey)`, ktorej prvým vstupom je vždy obrázok z kamery, ktorý bol v predošlej funkcií hranovaný a upravený do požadovaného tvaru vyplnenej kontúry a druhým vstupom pôvodný šedotónový obrázok.
V tele funkcie sú ako prvé vyhľadané všetky kontúry nachádzajúce sa v obraze pomocou funkcie `cv2.findContours(image, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)`. Funkcia vyhľadáva iba biele kontúry v obraze s čiernym pozadím, preto bol obraz na počiatku negovaný. Následne na základe podmienky sa kontroluje, či sa nachádza aspoň jedna kontúra v obraze. Ak áno, znamená to, že v inšpekčnej zóne pod kamerou sa nachádza objekt, alebo výraznejší šum pozadia. Po splnení podmienky je vyhľadaná najväčšia kontúra v obraze a na ňu sú

aplikované za sebou funkcie `cv2.minAreaRect(points)` a `cv2.boxPoints(rect)`, ktorých výstupom je objekt vo forme opísaného obdĺžnika okolo kontúry a pole so súradnicami vrcholov. Z objektu je následne vyčítaná dĺžka a šírka obdĺžnika, resp. objektu, počiatočné body *x, y* opísaného obdĺžnika a uhol natočenia obdĺžnika, resp. objektu. Na základe získaných znalostí o dĺžke a šírke objektu sa v následných podmienkach kontroluje podľa dopredu známych a definovaných veľkostí konektorov v pixeloch (podľa súboru `constants.py`), či sa jedná o čierny alebo hnedý/šedý konektor, ktoré sú predmetom inšpekcie. Ak áno, nastaví sa príznak detekcie konektoru a volá sa ďalšia funkcia, popísaná v ďalšej podkapitole. Ak nebola splnená podmienka prítomnosti aspoň jednej kontúry v obraze, alebo nájdená kontúra nezodpovedala rozmerom konektorov, vystúpi sa z funkcie bez výstupnej hodnoty a opäť sa volajú funkcie popísané vyššie.

7.2.3 Kompenzácia natočenia konektoru - CompAngle

Pre kompenzáciu natočenia uhla konektoru bola vytvorená funkcia `CompAngle(angleA, angleB, xCrop, yCrop, wCrop, hCrop, coordins, objWidth, objHeight, objAngle, imageReady, imageGrey)`, ktorej vstupné parametre je možné za radom popísať ako: uhol kompenzácie (získaný uhol natočenia opísaného obdĺžnika objekt navracia len v rozmedzí 0 - 90 stupňov, čiže v prípade natočenia obdĺžnika o záporný uhol treba navrátený uhol kompenzovať konštantou), súradnice počiatku orezania objektu, dĺžka a šírka objektu na orezanie, súradnice opísaného obdĺžnika a jeho dĺžka, šírka a uhol natočenia, hranovaný a šedotónový obrázok.
Výstupom z funkcie sú počiatočné body obdĺžnika opísaného okolo kompenzovaného konektoru na uhol 0 stupňov, ďalej dĺžka a šírka obdĺžnika a kompenzovaný šedotónový snímok, na ktorom sa následne bude vykonávať segmentácia výrobného čísla.
V tele funkcie je najskôr vysegmentovaný detegovaný konektor z pôvodného hranovaného a šedotónového obrázku. Následne sa porovnáva veľkosť dĺžky a šírky opísaného obdĺžnika zistených pomocou Euklidovskej vzdialenosti medzi jednotlivými vrcholmi. Takéto dynamické zisťovanie aktuálnej vzdialenosti medzi vrcholmi je potrebné, lebo použitá funkcia `cv2.boxPoints(rect)` má návratový typ pole o veľkosti 4, kde pod indexy 0 a 1 vždy ukladá dvojicu súradníc vrcholov, ktoré sú najviac napravo, čo je vlastnosť funkcie. S výhodou je ale táto vlastnosť využívaná, keďže vďaka nej po porovnaní vzdialeností medzi súradnicami pod indexmi 0 a 1 a indexmi 1 a 2 sa dá určiť čo je dĺžka a čo šírka obdĺžnika a tým pádom, či je obdĺžnik natočený pod kladným alebo záporným uhlom. Pomocou popísaného postupu bol vyriešený problém s problematickým implicitným rozsahom rotácie 0 - 90 stupňov navrátených z objektu v knižnici OpenCV.
Pomocou informácie o natočení obdĺžnika je následne pomocou matice rotácie M a

funkcie `cv2.warpAffine(image, M, (cols, rows))` hranovaný a šedotónový obrázok pootočený naspäť na uhol 0 stupňov. Vo vykompenzovanom hranovanom obrázku sa opäť vyhľadá najväčšia kontúra, čím je získaná opäť poloha konektoru a stanovia sa súradnice vysegmentovania oblasti s výrobným číslom konektoru. O samotnú segmentáciu sa stará už nasledujúca funkcia popísaná v nasledujúcej podkapitole. Ak nie je potreba kompenzácie uhlu natočenia, je možnosť funkciu jednoducho vypnúť v súbore `constants.py`.
Vytvorenie a aplikovanie popísanej vlastnej funkcie bolo nevyhnutné, keďže sa jedná o experimentálne rozpoznávanie znakov z konektorov, ktoré sú vkladané do inšpekčnej zóny ručne, čiže pri nesprávnom uložení konektoru pod uhlom by nastal problém pri aplikovaní Tesseractu na vysegmentovanú časť obrázku so znakmi. Vďaka tejto funkcii je vždy dodržaný 2. bod podmienok popísaný v kapitole 5.4.

7.2.4 Segmentácia výrobného čísla konektoru - CropPlateNum

Pre segmentáciu výrobného čísla konektoru s dostatočne veľkým okolím bola vytvorená funkcia `CropPlateNum(xPoint, yPoint, imageGrey, y1, y2, x1, x2)`. Do funkcie vstupujú za radom parametre: súradnice počiatočných bodov x,y opísaného obdĺžnika, šedotónový obrázok, súradnice potrebné pre segmentáciu výrobného čísla podľa aktuálne detegovaného konektoru, ktoré sú konštantami. Funkcia je triviálna, operácia vysegmentovania čísla zo šedotónového obrázku sa vykonáva priamo pri ukončení funkcie v návratovej hodnote.

7.2.5 Aplikovanie Tesseractu - ApplyTesseract

Pre získanie reťazca znakov z vysegmentovaného šedotónového obrázku bola vytvorená funkcia `ApplyTesseract(image, blocksize, const)`.Vstupným parametrami funkcie sú vysegmentovaný šedotónový obrázok výrobného čísla, hodnota prahu, upravovacia konštanta.
Pred samotným aplikovaním Tesseractu je vykonaných ešte niekoľko potrebných operácií s obrázkom pre jeho prípravu do vhodného tvaru. Obrázok je najskôr rozostretý pomocou funkcie spriemerovania `(src, ksize, dst, anchor, borderType)`, potom je na obrázok aplikované adaptívne prahovanie pomocou funkcie `cv.AdaptiveThreshold(src, dst, maxValue, adaptive_method=CV_ADAPTIVE_THRESH_MEAN_C, thresholdType=CV_THRESH_BINARY, blockSize, param1)`. Vďaka použitiu operácie spriemerovania pred adaptívnym prahovaním, je v prahovanom obrázku nižšie množstvo šumu pozadia. Následne je aplikovaná na prahovaný obrázok negácia, aby mali kontúry znakov a šum bielu farbu pixelov a pozadie čiernu farbu pixelov. Potom je opäť využité hľadanie kontúr v obrázku, ale v tomto prípade sú vyhľadané kontúry väčšie ako najmenší rozpoznávaný znak (pri čiernom konektore je to znak

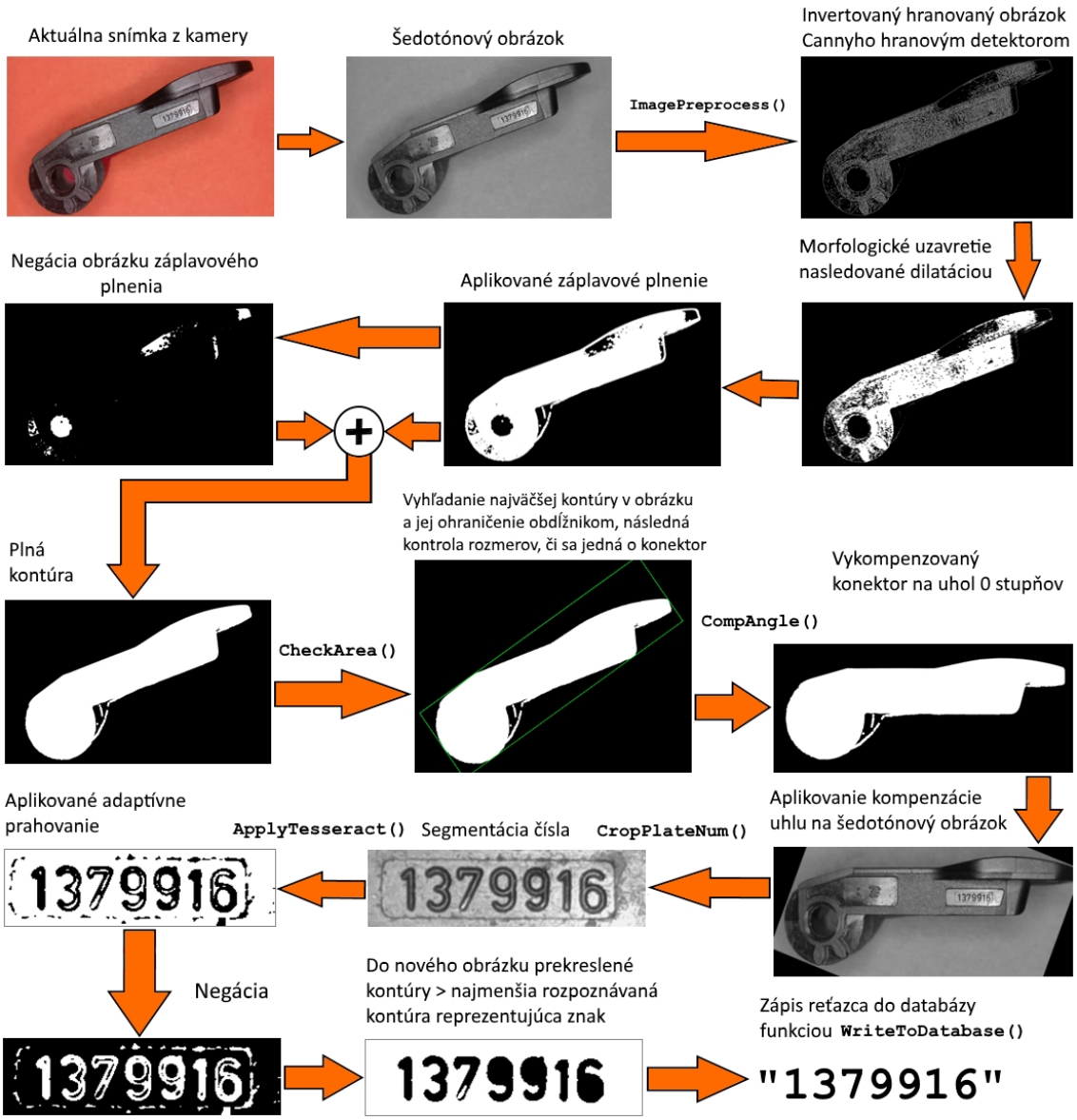
jednotky, pri hnedom/šedom je to znak pomlčky) a následne sú nájdené kontúry invertované do čiernej farby pixelov a vykreslené do nového obrázku o veľkosti pôvodného vysegmentovaného obrázku s bielou farbou pozadia. Týmto spôsobom je obrázok vhodne pripravený na aplikovanie Tesseractu, obrázok obsahuje minimálny, alebo žiadny šum pozadia a je tým pádom dodržaný 4. bod podmienok popísaný v kapitole 5.4. Nakoniec sa jednoducho na obrázok aplikuje Tesseract pomocou funkcie `pytesseract.image_to_string(image, config=")`, čím je získaný reťazec znakov výrobného čísla konektoru.

7.2.6 Zápis výsledkov do databázy - WriteToDatabase

Pre zápis reťazca znakov výrobného čísla konektoru do databázy bola vytvorená funkcia `WriteToDatabase(recgText)`, ktorá má vstupný parameter reťazec znakov výrobného čísla.
V tele funkcie je najskôr predpripravený reťazec s kľúčovým slovom označujúcim vkladanie do databázy spolu s názvami stĺpcov vytvorenej tabuľky databázy. Potom pomocou štandardnej štvorice príkazov zadaných za sebou v programe prebehne zápis do databázy a posledným z príkazov dôjde ku korektnému odpojeniu od databázy. Vytvorená databáza je popísaná v kapitole 5.5. Po úspešnom zapísaní do databázy program pokračuje naspäť na funkciu predspracovania snímkov a celý postup inšpekcie sa cyklicky opakuje v nekonečnej slučke.

7.2.7 Priebeh algoritmu rozpoznávania OCR

Na obrázku 7.1 sú ukázané názorne formou obrázkového diagramu všetky operácie vlastného inšpekčného programu za sebou, ktoré sa vždy vykonávajú na aktuálnej snímke z kamery. K ukážke poslúžila čierna súčiastka, rovnaké operácie sa ale vykonávajú aj pri zdetegovaní hnedého alebo šedého konektoru.



Obr. 7.1: Priebeh algoritmu inšpekcie výrobných čísel konektorov

7.3 Implementácia webovej vizualizácie

Ako už bolo spomenuté v kapitole 5.6.1, pre interakciu aplikácie rozpoznávania s užívateľom bola vytvorená webová vizualizácia. V nasledujúcich podkapitolách je popísaná implementácia výmeny informácií medzi hlavným programom rozpoznávania a programom zabezpečujúcim beh webovej vizualizácie zo strany serveru a samotná implementácia webovej vizualizácie zo strany klienta.

7.3.1 Predávanie reťazcov cez socket UDP klient-server

Odosielanie aktuálnych dát vo forme reťazcov obojsmerne medzi programami `recgApp.py` a `socketServer.py` bolo implementované pomocou socket UDP. Aktuálne dáta obsahujúce informácie o rozpoznaní konektoru, jeho výrobné číslo a stavové informácie sú odosielané z programu `recgApp.py` (v tomto prípade UDP klient) funkciou `SendServerData(data)` do programu `socketServer.py` (v tomto prípade UDP server). Na začiatok každej správy od klienta je vložený znak 'P' alebo 'L', podľa toho, či je správa určená vo webovej vizualizácii pre vyhodnotenie rozpoznania alebo informačné okno. Pre vytvorenie serverovej časti komunikácie bola v programe `socketServer.py` vytvorená trieda `SocketReciever`, ktorá pracuje v samostatnom vlákne. Trieda obsahuje metódu `run()`, v ktorej sa nachádza nadviazanie socket UDP komunikácie zo strany serveru s portom, na ktorom počúva. Po prijatí správy od klienta je kontrolovaný prvý znak správy, ak sa jedná o znak 'P' uloží sa reťazec so správou bez tohto prvého znaku do určenej globálnej premennej. Ak sa jedná o znak 'L', reťazec so správou bez tohto prvého znaku je uložený do inej určenej globálnej premennej.
Rovnako pre potreby odosielania informácie o zvolenom konektore vo webovej vizualizácii, na ktorom sa bude vykonávať inšpekcia, bola v programe `socketServer.py` (v tomto prípade UDP klient) vytvorená funkcia `SendClientData(connType)` pre odosielanie dát. V hlavnom programe `recgApp.py` (v tomto prípade UDP server) bola tiež vytvorená trieda `SocketReciever`, ktorá pracuje v samostatnom vlákne a obsahuje metódu `run()`, rovnako pre prijímanie správ. Po prijatí správy sú podľa reťazca s názvom druhu konektoru nastavené príslušné parametre prahovania pre vysegmentovaný obrázok výrobného čísla daného konektoru.
Predávanie správ ďalej webovej vizualizácii je popísané v nasledujúcej podkapitole. Popísané riešenie posielania dát medzi dvoma programami týmto spôsobom je potrebné, lebo odosielané hodnoty sú dynamické, čiže ich hodnoty sa počas behu programu menia. Spôsob získania dát, ktorý bol použitý medzi hlavným programom `recgApp.py` a programom s konštantami `constant.py`, ktorý spočíval v jednoduchom naimportovaní názvu programu s konštantami do hlavného programu, čím bol získaný prístup ku konštantám by v tomto prípade nefungoval.

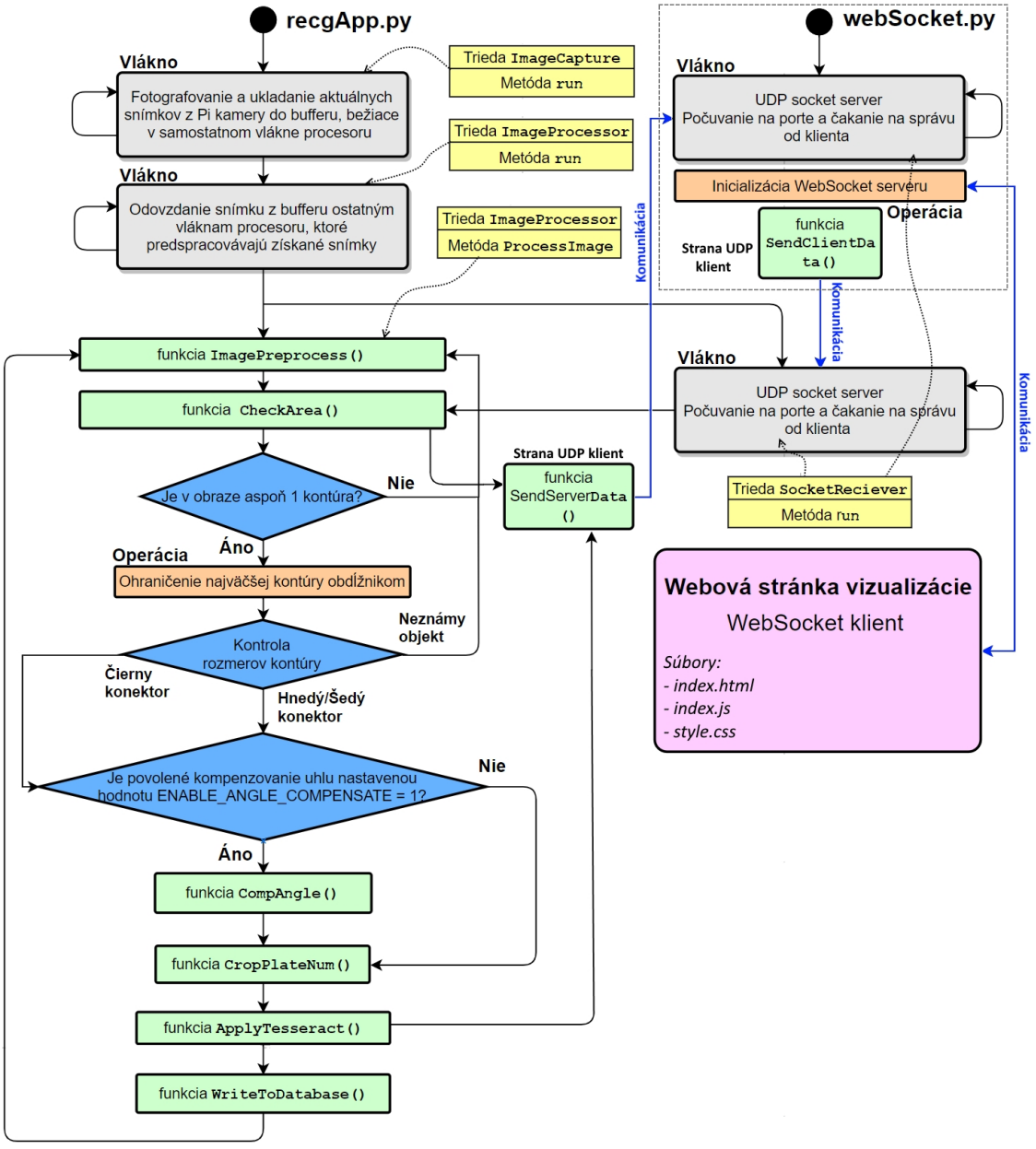
7.3.2 Implementácia WebSocket klient-server

Z hľadiska WebSocket server komunikácie je v programe *socketServer.py* inicializovaná táto komunikácia a nastavené previazanie so súborom `index.html`, ktorý je WebSocket klientom. Klient sa skladá z nasledujúcich súborov:

1. ***index.html*** - jedná sa o súbor v jazyku JavaScript, v ktorom sú deklarované všetky grafické prvky webovej stránky vizualizácie popísané v kapitole 8. Ďalej sa v ňom nachádza klientova časť WebSocket komunikácie s definovanou IP adresou s portom, na ktorú sa užívateľ pripája. Nachádza sa v ňom taktiež funkcia `setInterval()`, ktorá zabezpečuje načítanie aktuálnych hodnôt vyhodnotenia rozpoznania a správy pre informačné okno z programu *socketServer.py* pre webovú stránku vizualizácie každých 200 ms.
2. ***index.js*** - jedná sa o pomocný súbor v jazyku JavaScript, v ktorom sú implementované funkcie prvkov webovej vizualizácie. Jedna z funkcií konkrétne pri zmene prvku v rolete výberu konektoru odošle reťazec s názvom zvoleného konektoru WebSocket serveru. Ďalšia funkcia `addLog(logText)`, ktorá je automaticky volaná zo súboru *index.html* pri získaní aktuálneho reťazca pre informačné okno obohatí tento reťazec o časovú známku. Ďalšie funkcie volané zo súboru *index.html* sa starajú o aktualizovanie snímku rozpoznaného konektoru, vysegmentovaného šedotónového a prahovaného výrobného čísla každých 500 ms vo webovej vizualizácii.
3. ***style.css*** - jedná sa o súbor so štýlmi v jazyku CSS, ktorý sa stará o výzor každého prvku na webovej stránke vizualizácie, pozadie, umiestnenie prvkov na stránke a iné.

7.4 Priebeh celej inšpekčnej aplikácie

V nasledujúcom obrázku 7.2 je vývojový diagram celého inšpekčného programu aj s vizualizáciou. Sú v ňom vyznačené žltou farbou tri použité triedy s metódami spolu s odkazujúcimi čiarkovanými čiarami na vlákna, ktoré sú vytvorené v nich, v diagrame vyznačené šedou farbou. Ďalej sú zelenou farbou vyznačené elementy obsahujúce volania funkcií a modrou farbou rozhodovacie elementy. Oranžovou farbou sú operácie a samostatný fialový blok značí webové rozhranie.



Obr. 7.2: Vývojový diagram algoritmu inšpekcie výrobných čísel konektorov

8 GRAFICKÉ UŽÍVATEĽSKÉ ROZHRANIE

Pre interakciu s užívateľom bolo vytvorené webové grafické užívateľské rozhranie resp. webová vizualizácia (ďalej len GUI), ukázané na obrázku 8.1 pri jeho zobrazení v prehliadači Google Chrome. Obrázok obsahuje číselné označenie jednotlivých častí vizualizácie, za obrázkom nasleduje legenda s popisom. Na webové GUI je možné sa pripojiť automaticky hneď po spustení RPi.

Obr. 8.1: Grafické užívateľské rozhranie - popis jednotlivých častí

Popis častí GUI:

1. Názov web vizualizácie
2. Adresa pripojenia ku vizualizácii
3. Roleta so zoznamom objektov inšpekcie
4. Informačné okno
5. Rozpoznaný objekt
6. Vysegmentované výrobné číslo objektu v odtieňoch šedej
7. Vysegmentované prahované výrobné číslo objektu
8. Výsledok inšpekcie

Adresa pripojenia ku vizualizácii

Pre pripojenie sa k vizualizácii je potrebné zadať do okna aktuálnu IP adresu pridelenú pre RPi a vždy port `8080`, celý tvar vlastnej adresy z obrázku je `192.168.0.12:8080`.

Roleta so zoznamom objektov inšpekcie

Pred vložením konektoru do inšpekčnej zóny pod kamerou je potrebné, aby užívateľ zvolil z rolety možností, na ktorom druhu z konektorov ide vykonávať inšpekciu. Výber druhu konektoru slúži v hlavnom programe rozpoznávania pre exaktné nastavenie parametrov prahovania a podobne. Pre pokračovanie vykonávania inšpekcie na rovnakom druhu konektoru stačí ponechať zvolenú možnosť.

Informačné okno

Informuje užívateľa o aktuálnom stave rozpoznávania formou stavových správ. Každá stavová správa obsahuje časovú známku a samotnú správu. Správy popisujú zdetegovanie konektoru, rozpoznané číslo a informáciu o zápise do databázy.

Rozpoznaný objekt

Po vložení konektoru do inšpekčnej zóny a po jeho overení rozmerov je vo vizualizácii zobrazený aktuálny získaný snímok konektoru, na ktorom sa vykonáva inšpekcia výrobného čísla.

Vysegmentované výrobné číslo objektu v odtieňoch šedej

Vo vizualizácii je zobrazené taktiež samotné vysegmentované výrobné číslo konektoru, na ktorom sa vykonáva inšpekcia v odtieňoch šedej.

Vysegmentované prahované výrobné číslo objektu

Rovnako je vo vizualizácii vysegmentované prahované výrobné číslo bez šumu, na ktoré je aplikovaný Tesseract.

Výsledok inšpekcie

Nakoniec je rozpoznané výrobné číslo porovnané s referenčným výrobným číslom daného konektoru. Ak sa zhodujú, je vo vizualizácii vystavené veľkým zeleným písmom “OK”, čo značný dobrý kus. Ak nie, tak je vystavené veľkým červeným písmom “NOK”, čo značí zmätkový kus.

9 VYHODNOTENIE ÚSPEŠNOSTI INŠPEKCIE VÝROBNÝCH ČÍSEL KONEKTOROV

Pre overenie spoľahlivosti inšpekcie výrobných čísel konektorov bol každý z typov rozpoznávaných objektov vložený do inšpekčnej zóny pod kameru presne sto krát. Najskôr boli otestované všetky tri typy rozpoznávaných objektov s výrobnými číslami pri žiadnom resp. minimálnom natočení objektov pod uhlom položených kolmo pod kameru do inšpekčnej zóny. Dáta boli odčítané zo zapísaných spolu tristo riadkov v databázy - každý z troch typov objektov svojich sto riadkov rozlíšených pomocou stĺpca *Connector_type*, v ktorom bol vždy napísaný druh konektoru. Následne boli dáta spriemerované pomocou aritmetického priemeru. Tabuľka 9.1 obsahuje dosahované úspešnosti rozpoznávania výrobných čísel pri minimálnom natočení objektov. Ďalej boli otestované všetky tri druhy rozpoznávaných objektov s výrobným čís-

Typ rozpoznávaného objektu	**Úspešnosť rozpoznania výrobného čísla**
Čierna súčiastka	78,4 %
Hnedý konektor	76,1 %
Šedý konektor	75,6 %

Tab. 9.1: Úspešnosť rozpoznania výrobného čísla pri minimálnom natočení objektu o uhol

lom vložením každého z objektov do inšpekčnej zóny kolmo pod kameru sto krát, tentokrát ale pod ľubovolným uhlom až do hodnoty 180 stupňov. Tabuľka 9.2 obsahuje dosahované úspešnosti rozpoznávania výrobných čísel pri natočení objektov pod uhlom. Z tabuliek je patrné, že vyššej úspešnosti rozpoznávania bolo dosiahnuté

Typ rozpoznávaného objektu	**Úspešnosť rozpoznania výrobného čísla**
Čierna súčiastka	74,4 %
Hnedý konektor	72,8 %
Šedý konektor	71,2 %

Tab. 9.2: Úspešnosť rozpoznania výrobného čísla pri natočení objektu o uhol

pri uložení konektoru pod minimálnym uhlom. Pri natočení konektoru o veľký uhol sa zmenia svetelné podmienky pri rozložení svetla na konektore a výrobnom čísle, čo sa prejavuje zníženou úspešnosťou rozpoznania.
Najčastejším zdrojom chybného rozpoznania spôsobeného Tesseractom pri oboch spôsoboch testovania úspešnosti rozpoznávania bola zámena navzájom podobných znakov. Jednalo sa hlavne o zámenu medzi znakmi ’9’, ’0’ a ’6’. Pre účely testovania

bola nastavená rýchlosť FPS kamery na hodnotu 1, aby bol dostatok času na vloženie objektu do inšpekčnej zóny a eliminovali sa tak čo najviac rozmazané snímky, ktoré by figurovali v štatistike úspešnosti.
Z hľadiska rýchlosti rozpoznania výrobného čísla od zdetegovania objektu až po zápis rozpoznaného čísla s potrebnými informáciami do databázy sa časy pohybovali v intervale 3 až 4 sekúnd. Získané časy tým pádom spĺňajú základnú požiadavku vlastného experimentálneho rozpoznávania, ktorá bola stanovená do maxima 4 sekúnd.

Navrhnuté a implementované zariadenie inšpekcie výrobných čísel má obmedzenie v tom, že je vždy potrebné mechanické nastavenie presnej ohniskovej vzdialenosti objektívu Raspberry Pi kamery. Toto nastavenie je potrebné pri zmene vykonávania inšpekcie medzi čiernou súčiastkou a hnedým/šedým konektorom, keďže tieto objekty majú navzájom rôzne výšky uloženia výrobného čísla až o 1,6 cm, čím vznikajú rozdielne vzdialenosti od objektívu. Ďalším obmedzením zariadenia je aktuálne nastavená pevná výška objektívu vzdialeného od inšpekčnej plochy. V prípade zmeny tejto výšky zariadenie bude z hľadiska rozpoznávania pracovať nesprávne resp. s veľmi nízkou úspešnosťou rozpoznávania. Toto obmedzenie naväzuje na následné popísané možné vylepšenia práce.

Ak by bola aj naďalej udržaná základná myšlienka implementácie zariadenia rozpoznávania vedeného v experimentálnom duchu, vítaným vylepšením by mohlo byť umiestnenie kalibračnej značky o známych rozmeroch do inšpekčnej zóny, vďaka čomu by bol získaný skutočný údaj o veľkosti jedného pixelu v obraze v milimetroch a následne vďaka tomu by bolo možné dynamicky prepočítavať konštanty rozmerov konektorov, pomocou ktorých sa kontrolujú prítomné vložené objekty do inšpekčnej zóny. Ďalším vítaným vylepšením vlastného zariadenia rozpoznávania by mohlo byť predučenie neurónovej siete Tesseractu na používané dva druhy fontov výrobných čísel čiernej súčiastky a hnedého/šedého konektoru. Tým by sa ešte viac zvýšila úspešnosť rozpoznávania výrobných čísel.

Vytvorené zariadenie je pripravené na otestovanie v priemyselnom prostredí po splnení popísaných obmedzení resp. po kalibrácii. Celková cena vlastného experimentálneho zariadenia pre rozpoznávanie výrobných čísel bez započítania práce programátora, ktoré je od počiatku myslené ako nízko nákladové sa pohybuje okolo 120€. Cena vlastného zariadenia je neporovnateľne nižšia oproti komerčne používaným inšpekčným kamerám, ktoré stoja rádovo tisícky €.

10 ZÁVER

V tejto práci boli najskôr navrhnuté hardwarové prostriedky pre vlastnú aplikáciu strojového videnia. Pre implementáciu aplikácie bola zvolená firmou *TE Connectivity* platforma Raspberry Pi, konkrétne model Raspberry Pi 3B. Mikropočítač obsahuje všetky prostriedky od aplikácie rozpoznávania, cez webovú vizualizáciu, až po zapisovanie dát do databázy, teda aj s webovou vizualizáciou do nadradeného systému. Mikropočítač Raspberry Pi 3 Model B má vysoký výpočtový výkon, ktorý je možné rozložiť až do štyroch jadier, čo je žiaduce vo vlastnej aplikácii a naviac disponuje možnosťou bezdrôtového pripojenia do siete cez zabudovanú Wi-Fi anténu.

Návrh začal výberom vhodnej kamery s objektívom v kapitole 4.2, výber bol uskutočnený medzi klasickou web kamerou a Raspberry Pi kamerou. Keďže bola aplikácia vyvíjaná na platforme Raspberry Pi a hlavnou požiadavkou rozpoznania výrobných čísel bola rýchlosť rozpoznania, voľba bola jasná. Bola zvolená Raspberry Pi Camera(B), ktorá je menej známa, ale spĺňa rovnako všetky podmienky výberu ako viac známe typy RPi kamerové moduly, čo viac, disponuje nastaviteľným objektívom namontovaným v CS-mount objímke, ktorý prácu s kamerou a nastavenie zaostrovania obrazu v priestore podstatne uľahčuje. Kamera spĺňa očakávania aj čo sa týka kompatibility, kvality vyhotovených snímkov a ich nastaviteľnosti a rýchlosti prenosov dát z kamery do RPi.

V ďalšej kapitole 4.3 bol podrobný rozbor a porovnanie nasvietených výrobných čísel konektorov a súčiastky, pričom bol navrhnutý univerzálny osvetľovač, pre ktorý bolo u všetkých vzoriek možné s dostatočnou mierou opticky rozoznávať číslice, aby spĺňali podmienky pre úspešné rozpoznanie z obrazu. Je ním kruhový osvetľovač bielym LED svetlom, poskytnutý spoločnosťou *TE Connectivity*

V nasledujúcej kapitole 5 sa práca venovala inštalácii zvoleného operačného systému Raspbian Lite na Raspberry Pi a nevyhnutným nastaveniam mikropočítača pre interakciu s ním. V nadväzujúcich podkapitolách bol popísaný postup sprevádzkovania kamery a kompilácia a inštalácia knižnice OpenCV pre Python 3, ktorej funkcie boli využívané pri programovaní vlastnej aplikácie rozpoznávania. Dôležitými nasledujúcimi podkapitolami boli taktiež inštalácia engine Tesseractu pre optické rozpoznávanie, ktorý sprostredkováva samotné rozpoznanie textu z obrázku a nainštalovanie MariaDB serveru do Raspberry Pi, pre zapisovanie dát do databázy.

V následnej kapitole 6 bola navrhnutá jednoduchá mechanická konštrukcia prípravku a uloženia mikropočítača do spoločného púzdra s kamerovým modulom spolu s osvetľovačom, čím bolo vytvorené kompaktné zariadenie. Kompaktné zariadenie pripevnené na nosník mechanickej konštrukcie objektívom smerujúcim dole tak vytváralo inšpekčný prípravok s inšpekčnou zónou, do ktorej boli vkladané objekty,

ktoré boli predmetom inšpekcie.
V ďalšej kapitole 7 bol podrobne popísaný vytvorený algoritmus rozpoznávania so všetkými funkciami a názorným obrázkovým diagramom jednotlivých operácii a vysvetlená implementácia WebSocket komunikácie, ktorá sprostredkováva beh webovej vizualizácie.
V predposlednej kapitole 8, bolo popísané vytvorené webové grafické užívateľské prostredie z hľadiska používania užívateľom.
V poslednej kapitole bolo popísané zhrnutie dosiahnutých výsledkov s tým, že vlastné kompaktné zariadenie je funkčné a pripravené na testovanie.

Literatúra

[1] RUSNÁK, J.: *Návrh kamerového systému s průmyslovým robotem Kuka.* [online]. Vysoké učení technické v Brně. Fakulta strojního inženýrství, 2011 [cit. 2018-11-04]. Dostupné z: <http://hdl.handle.net/11012/1279>. Diplomová práce. Vysoké učení technické v Brně. Fakulta strojního inženýrství. Ústav výrobních strojů, systémů a robotiky. Vedoucí práce Tomáš Kubela.

[2] EPIC Systems, Inc.: *Quick History of Machine Vision.* [online]. 2018 [cit. 2018-11-04]. Dostupné z: <https://www.epicsysinc.com/blog/machine-vision-history>.

[3] Automa – časopis pro automatizační techniku, s.r.o.: *Strojové vidění I: Principy a charakteristiky.* [online]. 2008 [cit. 2018-11-07]. Dostupné z: <http://www.automa.cz/Aton/FileRepository/pdf_articles/36550.pdf>.

[4] Daily Automation, s.r.o.: *01 – Strojové videnie: Úvod (Machine Vision).* [online]. 2018 [cit. 2018-11-07]. Dostupné z: <http://dailyautomation.sk/01-strojove-videnie-uvod/>.

[5] Automa – časopis pro automatizační techniku, s.r.o.: *Strojové vidění II: Úlohy, nástroje a algoritmy.* [online]. 2008 [cit. 2018-11-07]. Dostupné z: <http://www.odbornecasopisy.cz/res/pdf/36676.pdf>.

[6] KNÁPKOVÁ, Eva.: *Aplikace systému strojového vidění.* [online]. Zlín: Univerzita Tomáše Bati ve Zlíně, 2012 [cit. 2018-11-07]. Dostupné z: <https://digilib.k.utb.cz/handle/10563/22385>. Bakalářská práce. Univerzita Tomáše Bati ve Zlíně. Fakulta aplikované informatiky, Ústav bezpečnostního inženýrství. Vedoucí práce Valouch, Jan.

[7] Automa – časopis pro automatizační techniku, s.r.o.: *Strojové vidění - několik úskalí návrhu systémů.* [online]. 2010 [cit. 2018-12-12]. Dostupné z: <http://www.mii.cz/download/company/articles/Automa_04_2010.pdf>.

[8] NATIONAL INSTRUMENTS: *A Practical Guide to Machine Vision Lighting.* [online]. 2017 [cit. 2018-12-13]. Dostupné z: <http://www.ni.com/white-paper/6901/en/>.

[9] GIEBL, Jan.: *Osvětlení ve strojovém vidění.* [online]. Plzeň: Západočeská univerzita v Plzni, 2016 [cit. 2018-12-13]. Dostupné z: <https://dspace5.zcu.cz/bitstream/11025/23184/1/BP_Giebl.pdf>. Bakalářská práce. Západočeská univerzita v Plzni. Fakulta elektrotechnická, Katedra aplikované elektroniky. Vedoucí práce Holota, Radek.

[10] KRAJCAR, Milan.: *Robotické vidění s průmyslovými roboty Kuka*. [online]. Vysoké učení technické v Brně. Fakulta strojního inženýrství, 2009 [cit. 2018-12-17]. Dostupné z: <http://hdl.handle.net/11012/539>. Diplomová práce. Vysoké učení technické v Brně. Fakulta strojního inženýrství. Ústav výrobních strojů, systémů a robotiky. Vedoucí práce Tomáš Kubela.

[11] AT␣&␣P journal: *Detekcia a sledovanie objektov(1)*. [online]. 2005 [cit. 2019-04-25]. Dostupné z: <https://www.atpjournal.sk/buxus/docs/atp-2005-06-69.pdf>.

[12] OpenCV: *Morphological Transformations*. [online]. 2019 [cit. 2019-04-25]. Dostupné z: <https://docs.opencv.org/trunk/d9/d61/tutorial_py_morphological_ops.html>.

[13] E. Šikudová, Z. Černeková, W. Benešová, Z. Haladová, and J. Kučerová: *Počítačové videnie Detekcia a rozpoznávanie objektov*. Praha: Wikina Praha, 2013. ISBN 978-80-87925-06-5.

[14] stackoverflow: *Fill the holes in OpenCV*. [online]. 2019 [cit. 2019-04-26]. Dostupné z: <https://stackoverflow.com/questions/1716274/fill-the-holes-in-opencv>.

[15] OpenCV: *Canny Edge Detection*. [online]. 2019 [cit. 2019-04-27]. Dostupné z: <https://docs.opencv.org/master/da/d22/tutorial_py_canny.html>.

[16] TOMORI, Zoltán, NIKOROVIČ, Matej: *Počítačové videnie v praxi*. [online]. 2016 [cit. 2019-04-27]. Dostupné z: <http://home.saske.sk/~tomori/Downloads/Poc_videnie/PV_2016.pdf>.

[17] VŠETIČKA, Václav.: *Digitální filtry pro obrazová data*. [online]. Vysoké učení technické v Brně. Fakulta elektrotechniky a komunikačních technologií, 2010 [cit. 2019-04-27]. Dostupné z: <https://www.vutbr.cz/www_basezav_prace_soubor_verejne.php?file_id=31349>. Bakalářská práce. Vysoké učení technické v Brně. Fakulta elektrotechniky a komunikačních technologií. Ústav telekomunikací. Vedoucí práce Petr Číka.

[18] LEŠKO, Vladimír.: *Mapování dopravního značení za pomoci metod zpracování obrazu*. [online]. Vysoké učení technické v Brně. Fakulta informačních technologií, 2015 [cit. 2019-04-27]. Dostupné z: <https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=115168>. Bakalářská práce. Vysoké učení technické v Brně. Fakulta informačních technologií. Ústav počítačové grafiky a multimédií. Vedoucí práce Michal Španěl.

[19] HARTMAN, Zdeněk.: *Modul pro rozpoznání nápisu pro robota*. [online]. Vysoké učení technické v Brně. Fakulta informačních technologií, 2015 [cit. 2019-04-27]. Dostupné z: <https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=115179>. Bakalářská práce. Vysoké učení technické v Brně. Fakulta informačních technologií. Ústav počítačových systémů. Vedoucí práce Tomáš Novotný.

[20] SONKA, Milan, HLAVAC, Vaclav a BOYLE, Roger: *Image processing, analysis, and machine vision*. Fourth edition. Stamford: Cengage Learning, 2015. ISBN 978-11-33593-60-7.

[21] WIKIPEDIA: *Charge-coupled device*. [online]. 2018 [cit. 2018-12-17]. Dostupné z: <https://en.wikipedia.org/wiki/Charge-coupled_device>.

[22] WIKIPEDIA: *CMOS*. [online]. 2018 [cit. 2018-12-17]. Dostupné z: <https://en.wikipedia.org/wiki/CMOS>.

[23] HOTAŘ, Vlastimil. *Úvod do problematiky strojového vidění. Část 1, Základní principy a hardware*. Liberec: Technická univerzita v Liberci, 2015. ISBN 978-80-7494-156-6.

[24] DRONESGLOBE: *CCD VS CMOS SENSORS*. [online]. 2018 [cit. 2018-12-23]. Dostupné z: <http:/www.dronesglobe.com/guide/4k-drones/attachment/ccd-vs-cmos-sensors/>.

[25] DEMANT, Christian, Carsten GARNICA a Bernd STREICHER-ABELl. *Industrial image processing: Visual quality control in manufacturing*. Berlin: Springer Berlin Heidelberg, 2013. ISBN 978-36-4233-905-9.

[26] MEGAPIXEL s.r.o.: *Velikost snímače*. [online]. 2018 [cit. 2018-12-23]. Dostupné z: <https://www.megapixel.cz/velikost-snimace>.

[27] JENČ, Jiří: *Možnosti rozšíření Raspberry Pi o modul kamery*. [online]. Praha: České vysoké učení technické v Prahe, 2014 [cit. 2018-12-28]. Dostupné z: <http://users.fs.cvut.cz/ivo.bukovsky/PVVR/prace_studentu/Jenc_RPi_kamery_s_FFT.pdf>. Semestrálna práca.

[28] eLinux : *RPi USB Webcams*. [online]. 2018 [cit. 2018-12-28]. Dostupné z: <https://elinux.org/RPi_USB_Webcams>.

[29] DIY ELECTRONICS: *LIST OF PRODUCTS BY MANUFACTURER WAVESHARE*. [online]. 2018 [cit. 2018-12-28]. Dostupné z: <https://www.diyelectronics.co.za/store/9_waveshare>.

[30] SGBotic Pte Ltd.: *Raspberry Pi Camera Module w/Adjustable Focus Lens*. [online]. 2018. Dostupné z: <https://www.sgbotic.com/index.php?dispatch=products.view&product_id=1941>.

[31] OmniVision: *Datasheet: OV5647*. [online]. 2009 [cit. 2018-12-29]. Dostupné z: <https://cdn.sparkfun.com/datasheets/Dev/RaspberryPi/ov5647_full.pdf>.

[32] Cours d'ordinateur: *Raspberry setup*. [online]. 2019. Dostupné z: <http://www.cours-ordinateur.fr/raspberry-setup/>.

[33] SocialCompare: *RaspberryPI models comparison*. [online]. 2019 [cit. 2018-12-30]. Dostupné z: <http://socialcompare.com/en/comparison/raspberrypi-models-comparison>.

[34] Opto Engineering: *PRODUCT DATASHEET: RT-LSW-15-050-2-W-24V-FL*. [online]. 2019. Dostupné z: <https://www.opto-e.com/media/pdf/RT-LSW-15-050-2-W-24V-FL-datasheet-en.pdf>.

[35] Opto Engineering: *PRODUCT DATASHEET: RT-DLR2-60-100-2-W-24V-FL*. [online]. 2019. Dostupné z: <https://www.opto-e.com/media/pdf/RT-DLR2-60-100-2-W-24V-FL-datasheet-en.pdf>.

[36] Opto Engineering: *PRODUCT DATASHEET: RT-LBRX-00-120-6-W-24V-FL*. [online]. 2019. Dostupné z: <https://www.opto-e.com/media/pdf/RT-LBRX-00-120-6-W-24V-FL-datasheet-en.pdf>.

[37] WIKIPEDIA: *Raspberry Pi*. [online]. 2019 [cit. 2019-01-06]. Dostupné z: <https://en.wikipedia.org/wiki/Raspberry_Pi>.

[38] Safari Books Online: *History of OpenCV from v1 to v4*. [online]. 2019 [cit. 2019-05-01]. Dostupné z: <https://www.oreilly.com/library/view/mastering-opencv-4/9781789533576/2de0893d-e450-417d-b336-a4143799b43d.xhtml>.

[39] BALCI, Batuhan, SAADATI, Dan, SHIFERAW, Dan: *Handwritten Text Recognition using Deep Learning*. [online]. 2017 [cit. 2019-05-01]. Dostupné z: <http://cs231n.stanford.edu/reports/2017/pdfs/810.pdf>.

[40] Tecmint: Linux Howtos, Tutorials &␣Guides: *The Story Behind Acquisition of 'MySQL' by Sun Microsystem and the Rise of 'MariaDB'*. [online]. 2019 [cit. 2019-05-01]. Dostupné z: <https://www.tecmint.com/the-story-behind-acquisition-of-mysql-and-the-rise-of-mariadb/>.

[41] PiBorg: *Faster video processing*. [online]. 2019 [cit. 2019-05-02]. Dostupné z: <http://forum.piborg.org/node/2363>.

[42] websocket.org: *Faster video processing*. [online]. 2019 [cit. 2019-05-03]. Dostupné z: <http://www.websocket.org/aboutwebsocket.html>.

[43] python-socketio: *Getting Started*. [online]. 2019 [cit. 2019-05-03]. Dostupné z: <https://python-socketio.readthedocs.io/en/latest/intro.html#what-is-socket-io>.

Zoznam symbolov, veličín a skratiek

CCD	Charge Coupled Device
1D	One Dimensional
2D	Two Dimensional
MMI	Man Machine Interface
CMOS	Complementary Metal Oxide Semiconductor
A/D	Analog/Digital
Mpx	Mega pixel
OCR	Optical Character Recognition
USB	Universal Serial Bus
HDMI	High Definition Multimedia Interface
RCA	Radio Corporation of America
LCD	Liquid Crystal Display
DSI	Display Serial Interface
GPIO	General Purpose Input Output
UART	Universal Asynchronous Receiver Transmitter
SPI	Serial Peripheral Interface
CSI	Camera Serial Interface
SD	Secure Digital
Wi-Fi	Wireless-Fidelity
Mb/s	Mega bits per second
Gb/s	Giga bits per second
HD	High Definition
FPS	Framerate Per Second
I2C	Inter Integrated Circuit
LED	Light Emitting Diode
OS	Operation System
IP	Internet Protocol
SSH	Secure Shell
GUI	Graphic User Interface
VNC	Virtual Network Computing
SFTP	Secure File Transfer Protocol
SCP	Secure copy protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

Zoznam príloh